USING ADVANCED COMPUTING IN APPLIED DYNAMICS: FROM THE DYNAMICS OF GRANULAR MATERIAL TO THE MOTION OF THE MARS ROVER

Dan Negrut
NVIDIA CUDA Fellow
Simulation-Based Engineering Lab
Wisconsin Applied Computing Center
University of Wisconsin-Madison

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
01 FEB 2014	**Briefing Charts**	**14-11-2013 to 05-01-2014**

4. TITLE AND SUBTITLE
USING ADVANCED COMPUTING IN APPLIED DYNAMICS: FROM THE DYNAMICS OF GRANULAR MATERIAL TO THE MOTION OF THE MARS ROVER

5a. CONTRACT NUMBER
W911NF-11-F-0001-0229

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Dan Negrut

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Wisconsin Applied Computing Center,University of Wisconsin-Madison,500 Lincoln Dr,Madison,WI,53706

8. PERFORMING ORGANIZATION REPORT NUMBER
; #24389

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army TARDEC, 6501 East Eleven Mile Rd, Warren, Mi, 48397-5000

10. SPONSOR/MONITOR'S ACRONYM(S)
TARDEC

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
#24389

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Briefing Charts

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**Public Release**	**114**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

Acknowledgements: People

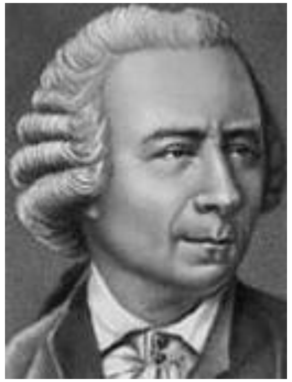
- Alessandro Tasora, University of Parma, Italy
- Drs. Paramsothy Jayakumar & David Lamb, US Army TARDEC
- Mihai Anitescu, University of Chicago & Argonne National Lab
- Jeff Freeman
- Paul Ayers, University of Tennessee

- Lab members, listed alphabetically:
  - Aaron Bartholomew, Luning Fang, Toby Heyn, Ang Li, Naresh Khude, Justin Madsen, Hammad Mazhar, Dan Melanz, Francisco Mercado, Spencer O'Rourke, Arman Pazouki, Andrew Seidl, Radu Serban, Nadia Sweet

Acknowledgements: Funding Sources

- National Science Foundation
- US Army Research Office (ARO)
- US Army TARDEC
- NVIDIA
- Simertis GmbH
- MSC.Software
- FunctionBay, S. Korea
- Caterpillar

Classical Computational Multibody Dynamics: Newton-Euler Constrained Equations of Motion



Generalized Positions

Generalized Mass Matrix

Velocity Transformation Matrix

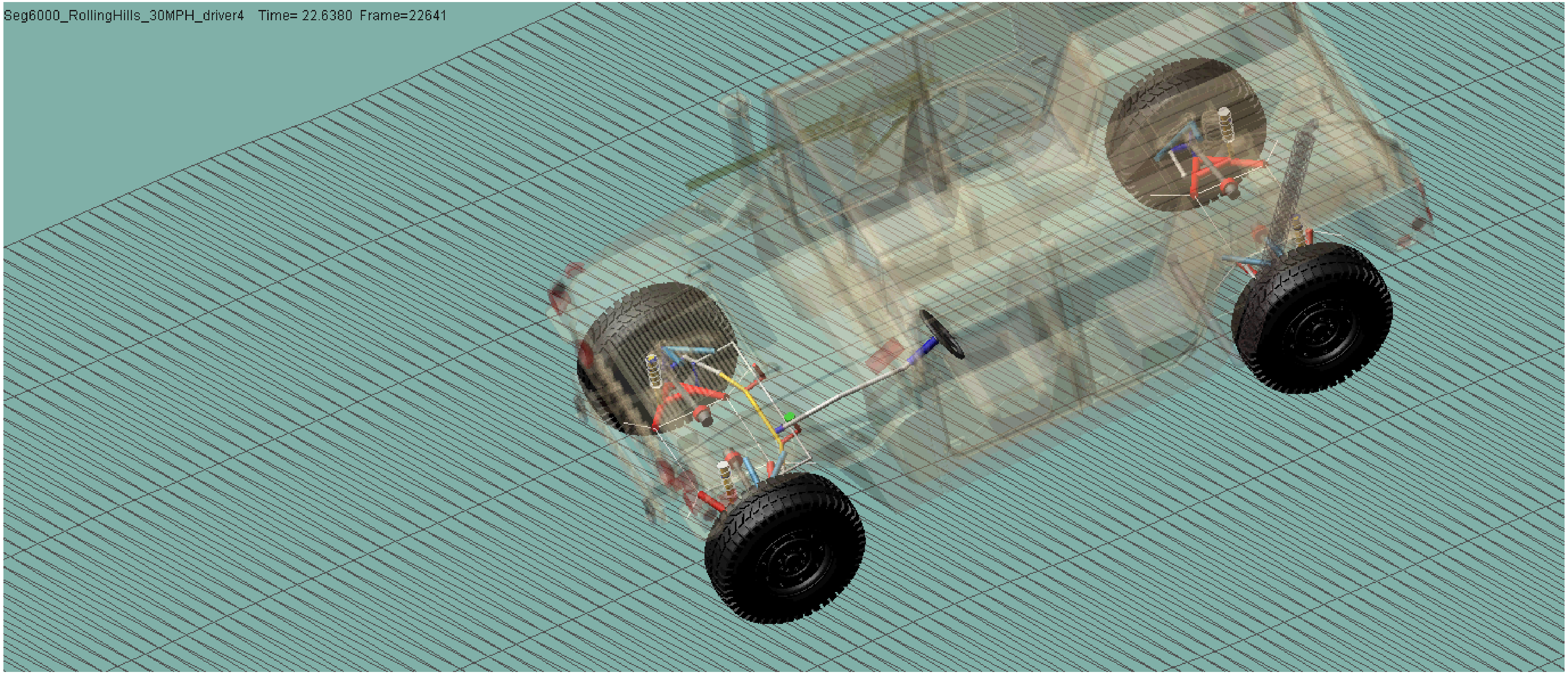
Generalized Velocities

Kinematic Differential Equations → $\dot{\mathbf{q}} = \mathbf{L}(\mathbf{q})\mathbf{v}$

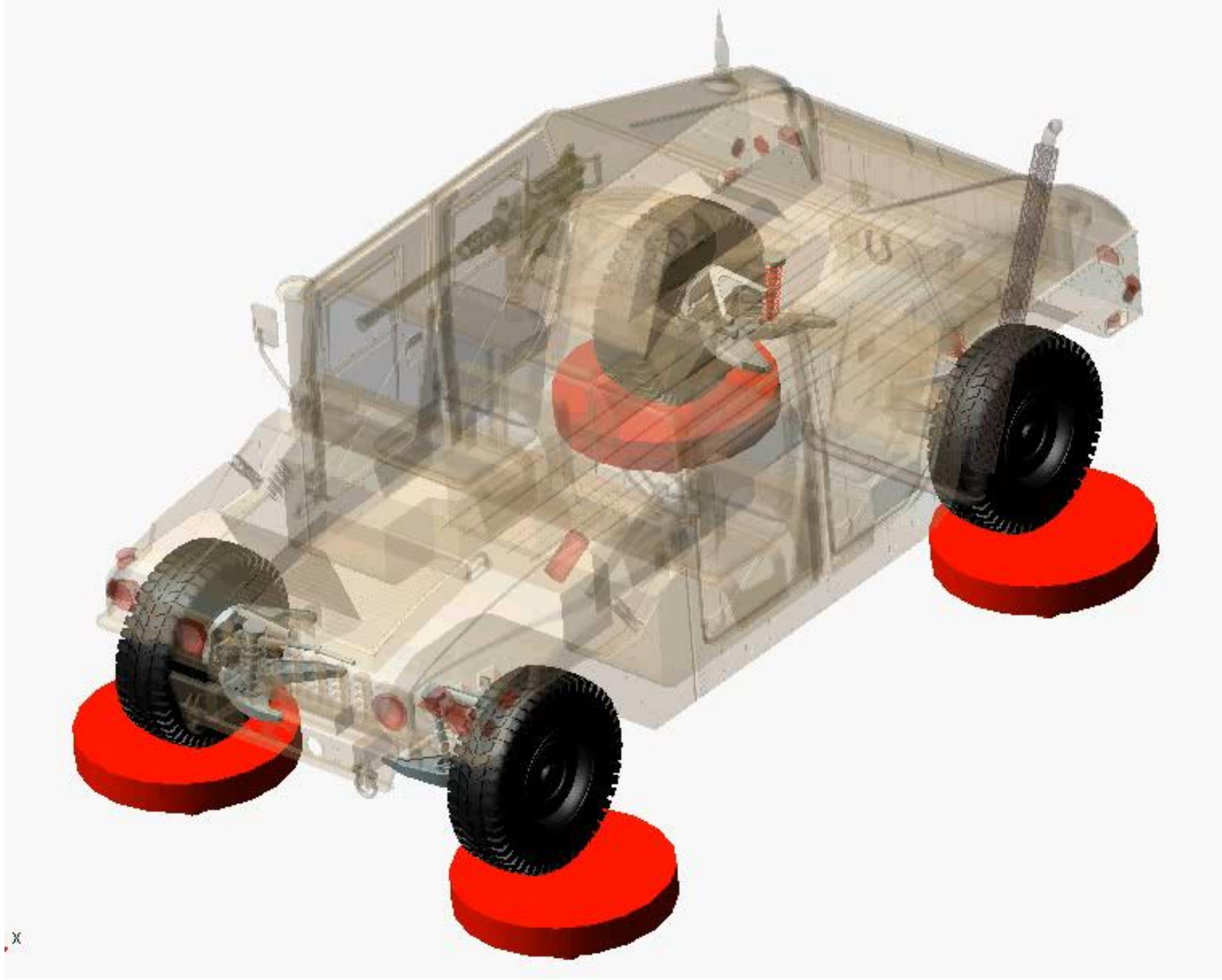
Force Balance Equations → $\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} = \underbrace{\mathbf{f}(t,\mathbf{q},\mathbf{v})}_{\text{Applied Force}} - \overbrace{\mathbf{g}_{\mathbf{q}}^{T}(\mathbf{q},t)\boldsymbol{\lambda}}^{\text{Reaction Force}}$

Holonomic Kinematic Constraints → $\mathbf{g}(\mathbf{q},t) = \mathbf{0}$

Multibody Dynamics: What is it? [commercial software simulation]



Multibody Dynamics: What is it? [commercial software simulation]

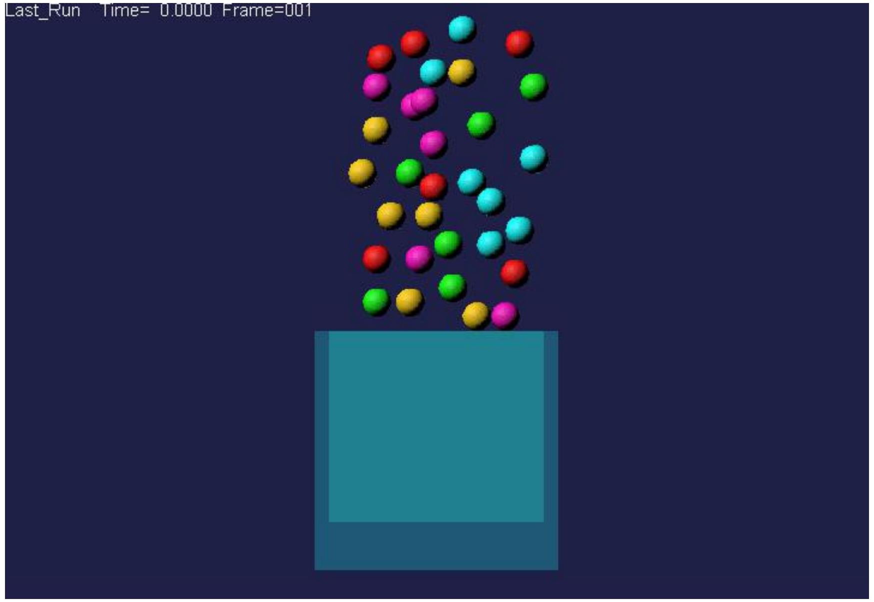


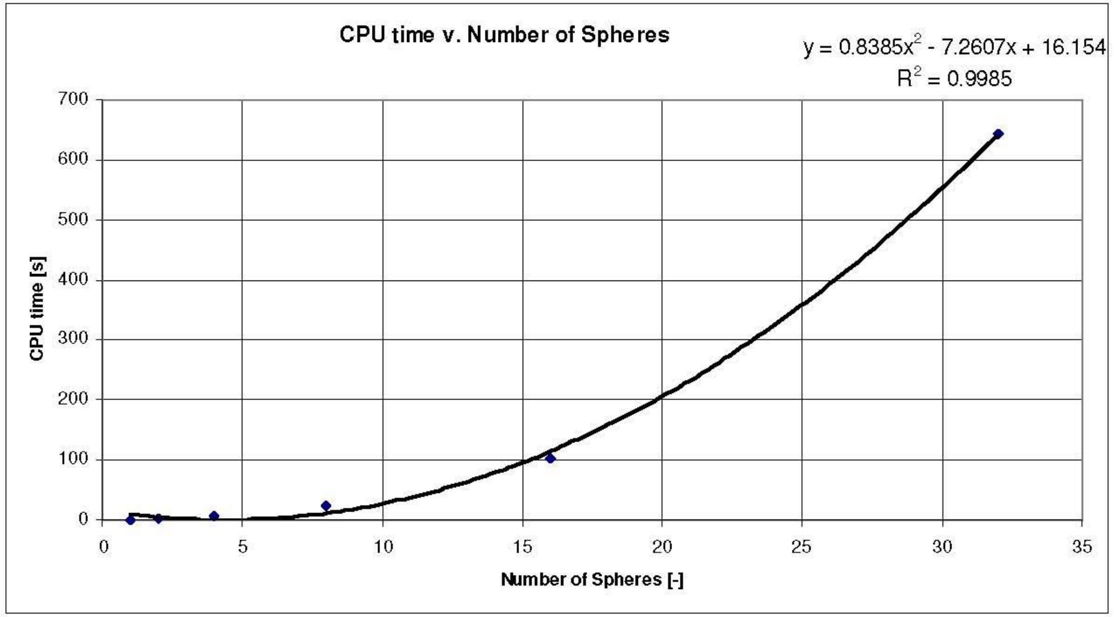
Example Open Problem: Mobility on Deformable Terrain

- How is the Rover moving along on a slope with granular material?
- What wheel geometry is more effective?
- How much power is needed to move it?
- At what grade will it get stuck?
- And so on…

Frictional Contact Simulation [Commercial Software Simulation - 2007]

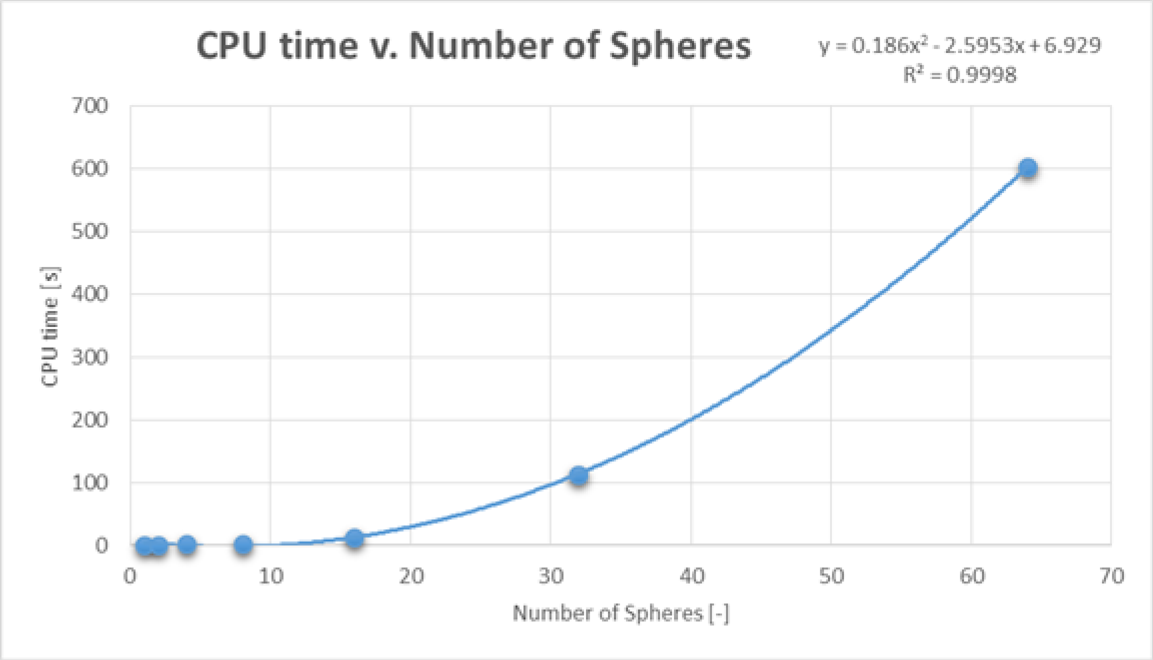
- Model Parameters:
  - Spheres: 60 mm diameter and mass 0.882 kg
  - Penalty Approach: stiffness of 1E5, force exponent of 2.2, damping coefficient of 10.0
  - Simulation length: 3 seconds



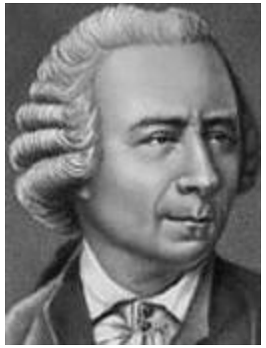


Frictional Contact Simulation [Commercial Software Simulation - 2013]

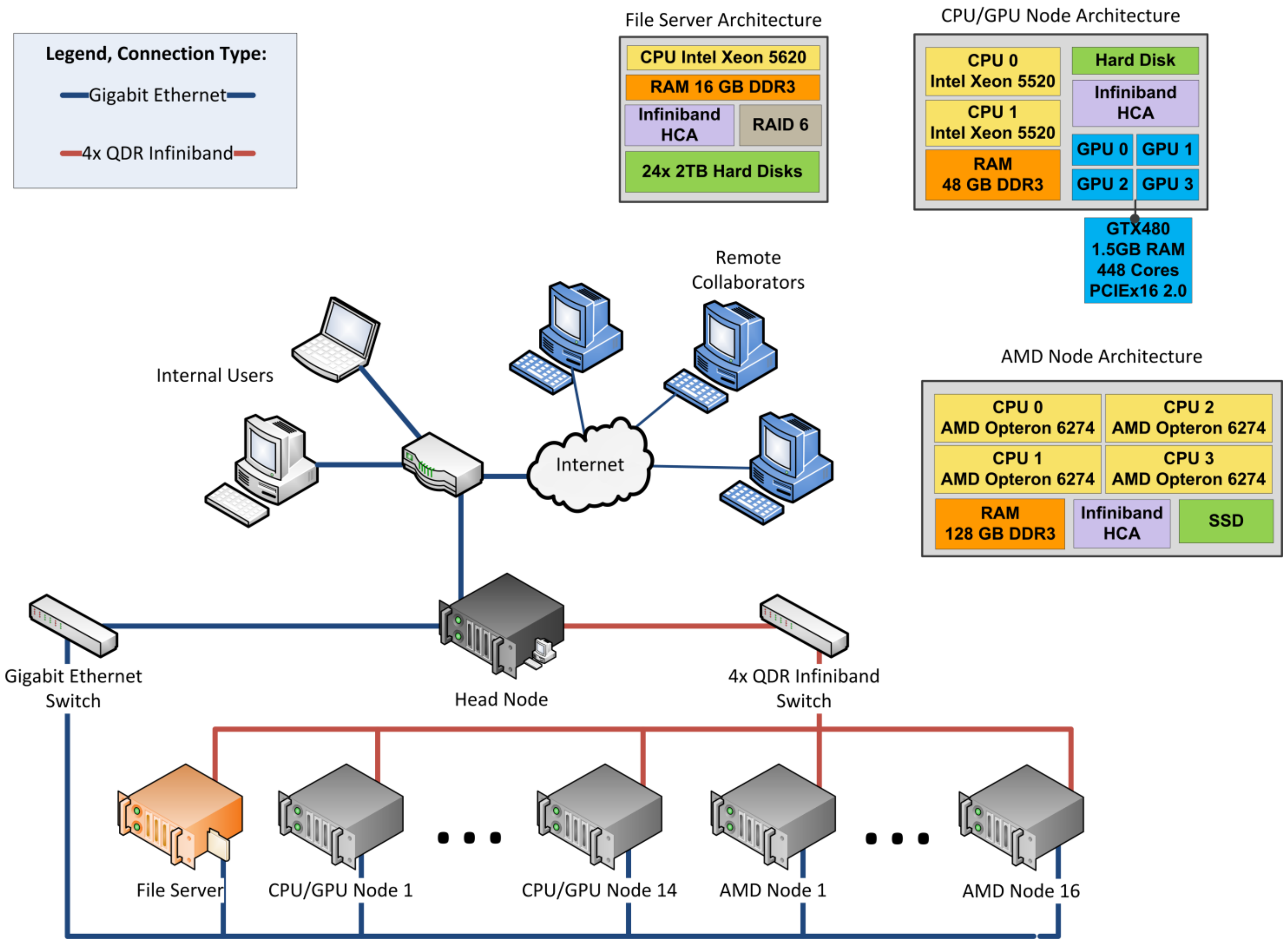
- Same problem tested in 2013
- Simulation time reduced by a factor of six
- Simulation times still prohibitively long



Why It's Worth Reconsidering Challenging Problems…



Lab's Research Heterogeneous Cluster



Lab's Research Heterogeneous Cluster

- More than 50,000 GPU scalar processors
- More than 1,200 CPU cores
- Fast Mellanox Infiniband Interconnect (QDR), 40Gb/sec
- About 2.7 TB of RAM
- More than 20 Tflops Double Precision

The issues is not hardware availability. Rather, it is producing modeling and solution techniques that can leverage the hardware

CHRONO:

Research-Grade Software Infrastructure for Multi-physics Modeling/Simulation/Visualization

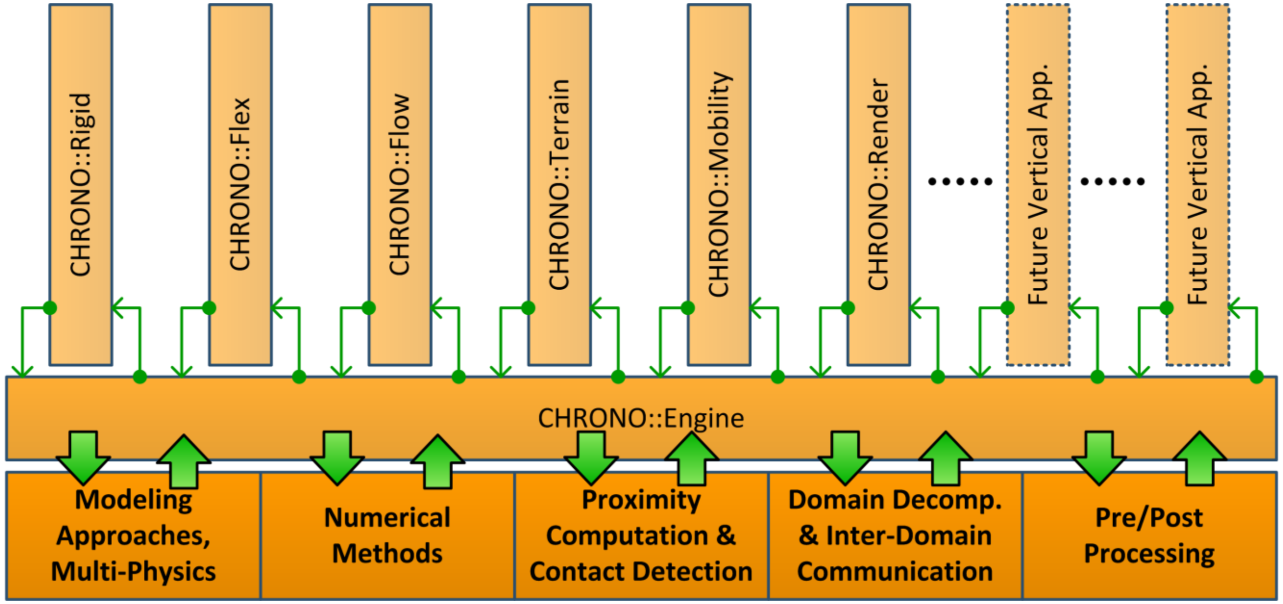
- Goal: advance state of the art in modeling, simulation, and visualization
  - Use emerging hardware and novel algorithms to solve open engineering problems

- "emerging hardware" :
  - GPUs and clusters of CPUs

- "open engineering problems" :
  - Fluid-solid interaction, vehicle mobility, soil modeling, tire/terrain modeling, granular dynamics, etc.

Chrono:
Five Foundation Components

- Advanced modeling
- Solution methods
- Proximity computation
- Domain decomposition & Inter-domain data exchange
- Pre/Post-processing (visualization)



- **Chrono**:
  - Five foundation components support vertical apps

Advanced Modeling Techniques

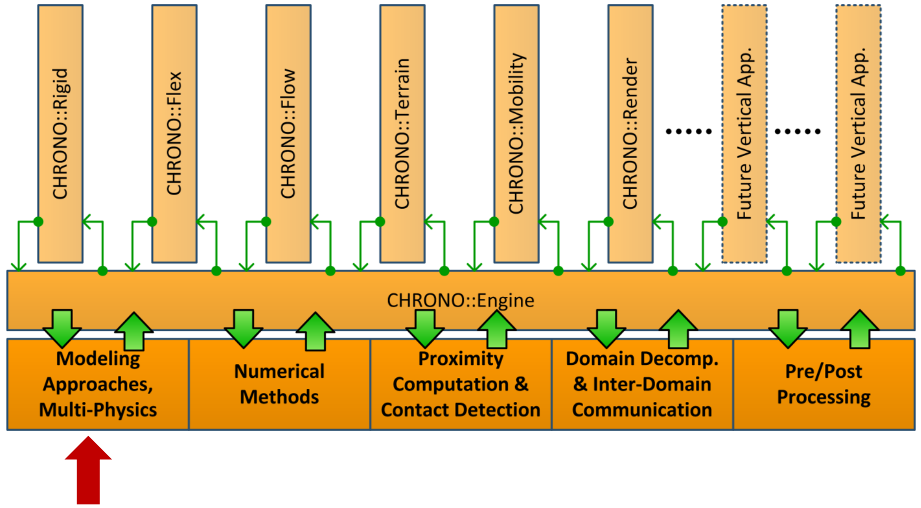
Advanced modeling techniques

Algorithmic (applied math) support

Proximity computation

Domain decomposition & Inter-domain data exchange

Post-processing (visualization)

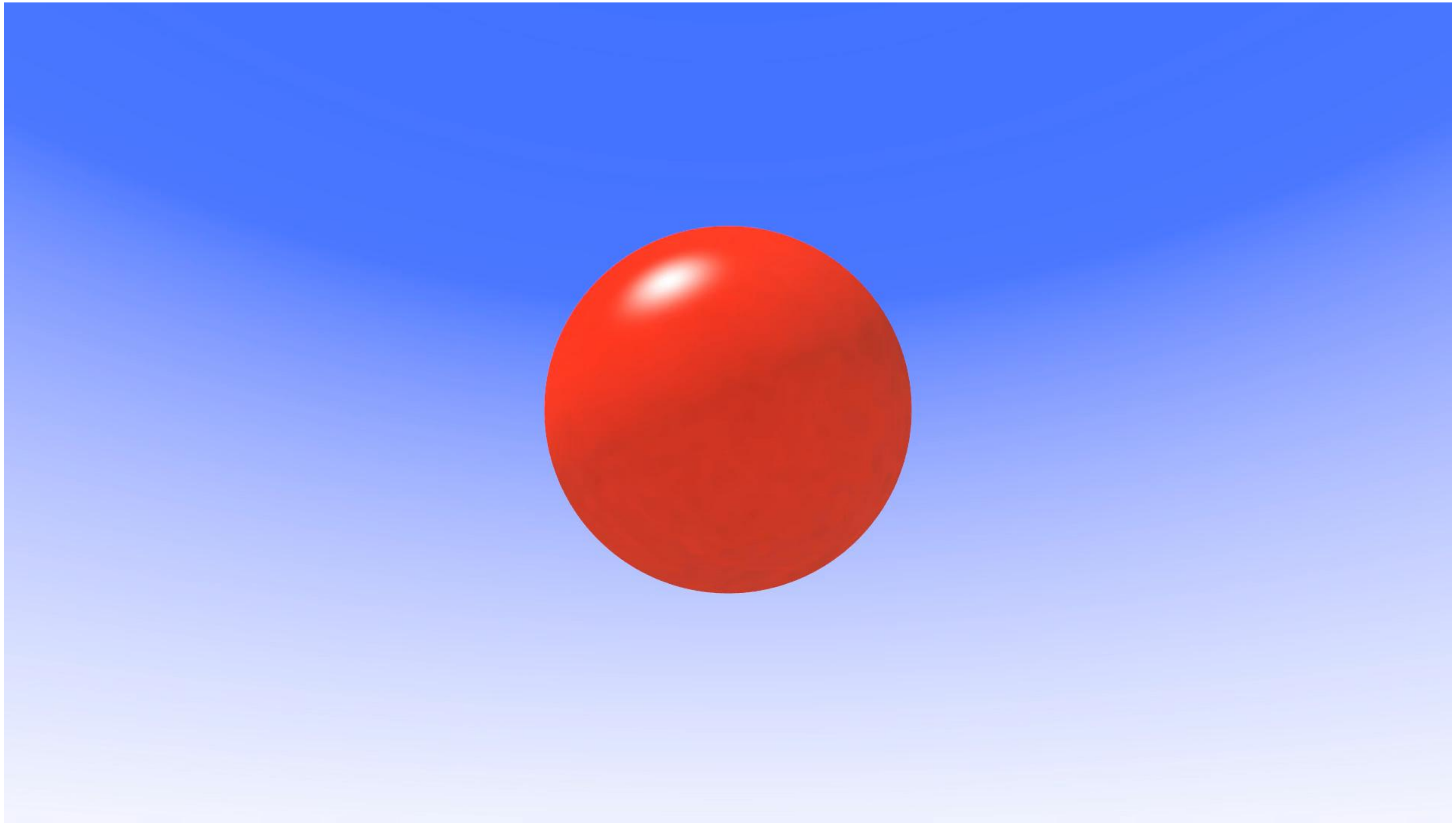


Chrono: Support for Advanced Modeling Techniques

- Modeling; what does it mean?
  - The process of formulating a set of governing differential equations that captures the physics associated with the engineering problem of interest

- Modeling decisions are consequential
  - Hallmark of good modeling: it leads to a palatable math problem that can be solved numerically with relative ease

Chrono::Flex - Dealing with Compliant Bodies



Deformable Body Modeling Support in **Chrono**

Equation of Motion & Mass Matrix

- Equations of Motion:

$$\mathbf{M}\ddot{\mathbf{e}} + \mathbf{Q}_s = \mathbf{Q}_e$$

- Mass matrix is constant and SPD

$$\mathbf{M} = \left[\int_{V_o} \rho \mathbf{S}^{\mathrm{T}} \mathbf{S} \, dV_o \right]$$

System Forces

- Due to gravity

$$\mathbf{Q}_e = A \int_0^l \mathbf{S}^T \mathbf{f}_g \, dx$$

- Due to a concentrated force:

$$\mathbf{Q}_e = \mathbf{S}^T \mathbf{f}$$

Deformable Bodies: Internal Forces…

- Strain Energy (shown for beam elements):

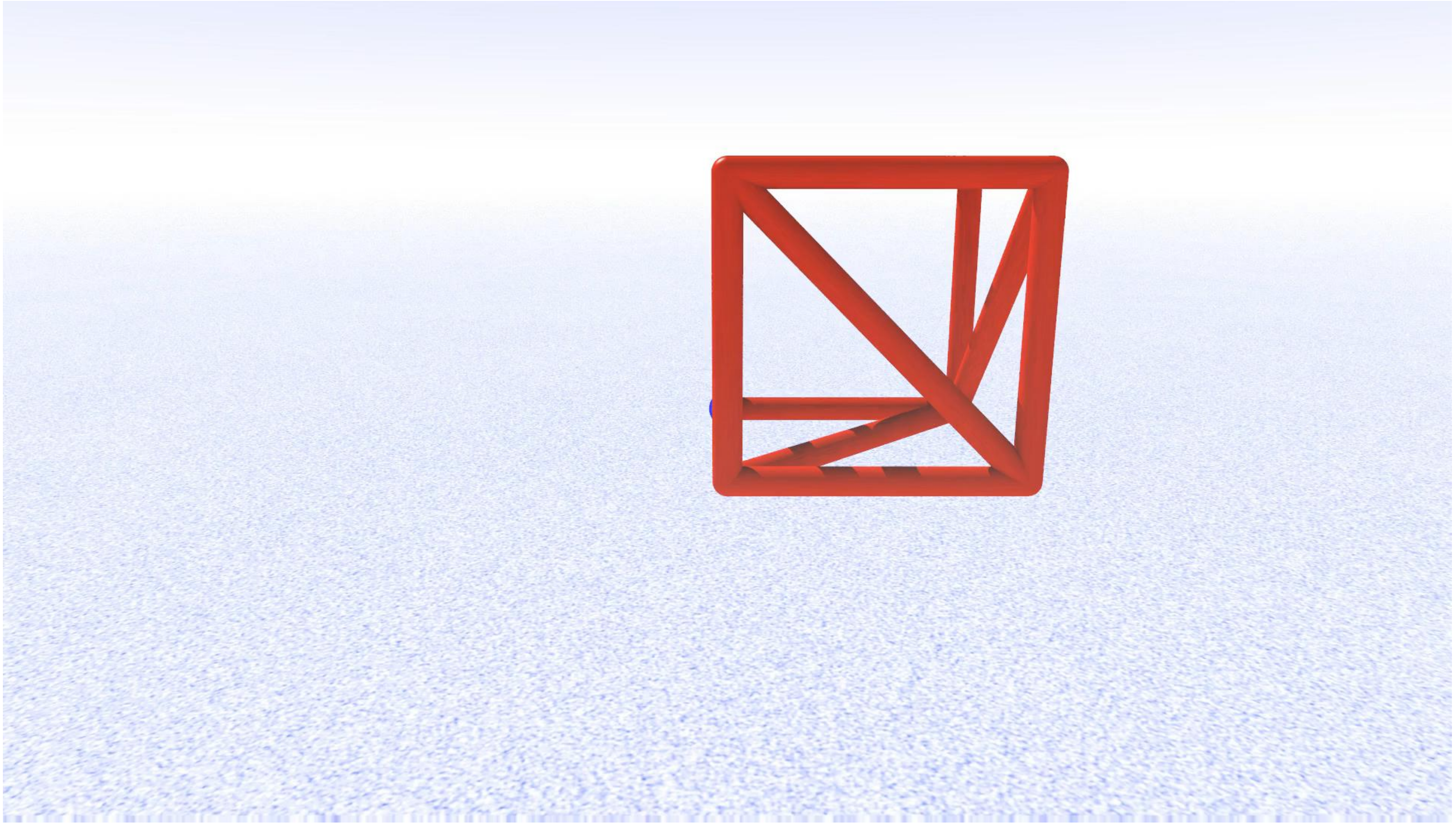
$$U = \frac{1}{2}\int_0^l EA(\varepsilon_{11})^2\,dx + \frac{1}{2}\int_0^l EI(\kappa)^2\,dx$$

- Partial Derivative of Strain Energy wrt generalized coordinated **e** yields the Internal Forces

$$\mathbf{Q}_s = \int_0^l EA(\varepsilon_{11})\left(\frac{\partial \varepsilon_{11}}{\partial \mathbf{e}}\right)^T dx + \int_0^l EI(\kappa)\left(\frac{\partial \kappa}{\partial \mathbf{e}}\right)^T dx$$

- Bad news: internal force expensive to evaluate
- Good News: for large systems can be done in parallel

Deformable Bodies with Constraints



Deformable Bodies with Constraints

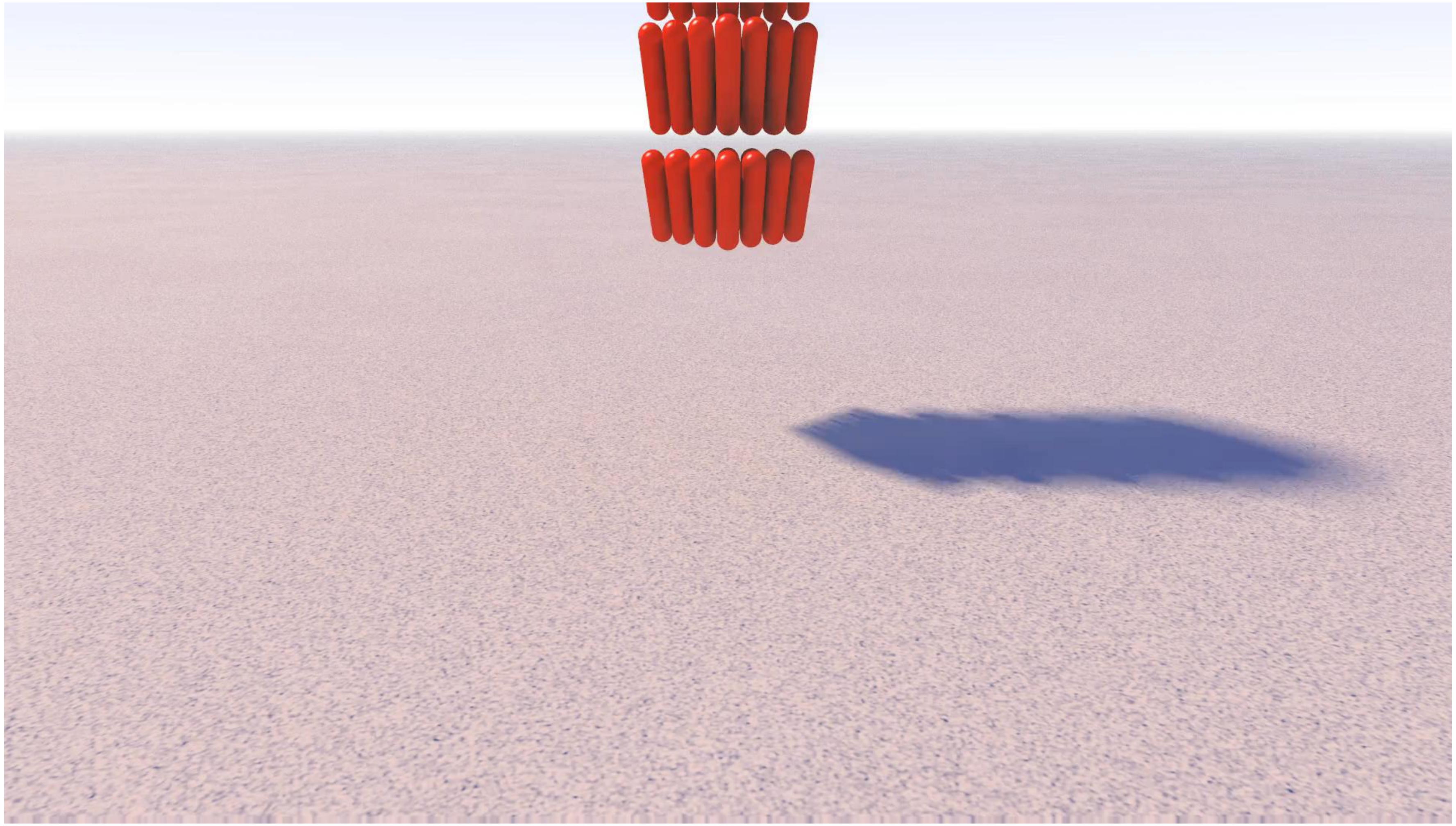
- Constraints assumed holonomic, formulated as

$$\mathbf{\Phi}(\mathbf{q},t) = [\Phi_1(\mathbf{q},t)\ldots\Phi_m(\mathbf{q},t)]^T = 0$$

- Constraint form of the Equations of Motion (index 3 DAE problem)

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{\Phi}_{\mathbf{q}}^T(\mathbf{q},t)\lambda + \mathbf{Q}_{\text{int}}(\mathbf{q}) = \mathbf{Q}_{\text{ext}}(\dot{\mathbf{q}},\mathbf{q},t)$$

Wiggly Bodies
[Flexible bodies, w/ Friction and Contact: parallel simulation on the GPU]

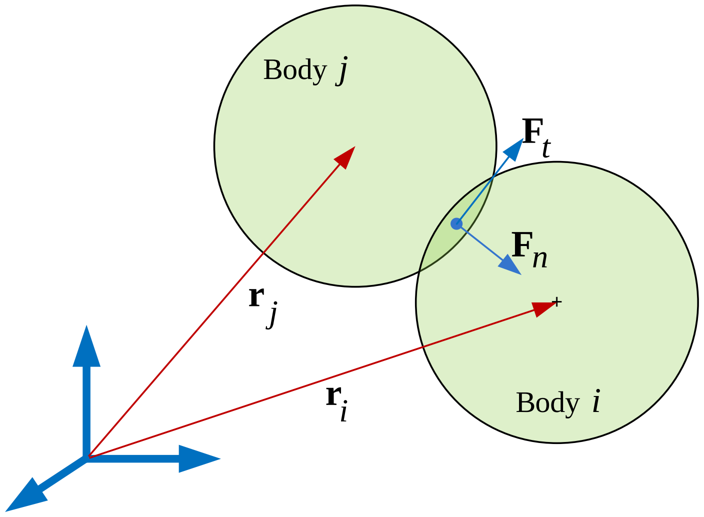


Deformable Bodies with Friction and Contact

- Contact forces depend on several parameters:
  - Contact penetration
  - Normal vector
  - Relative velocity of colliding bodies at the point of contact
  - Etc.
- Normal force due to a collision calculated as

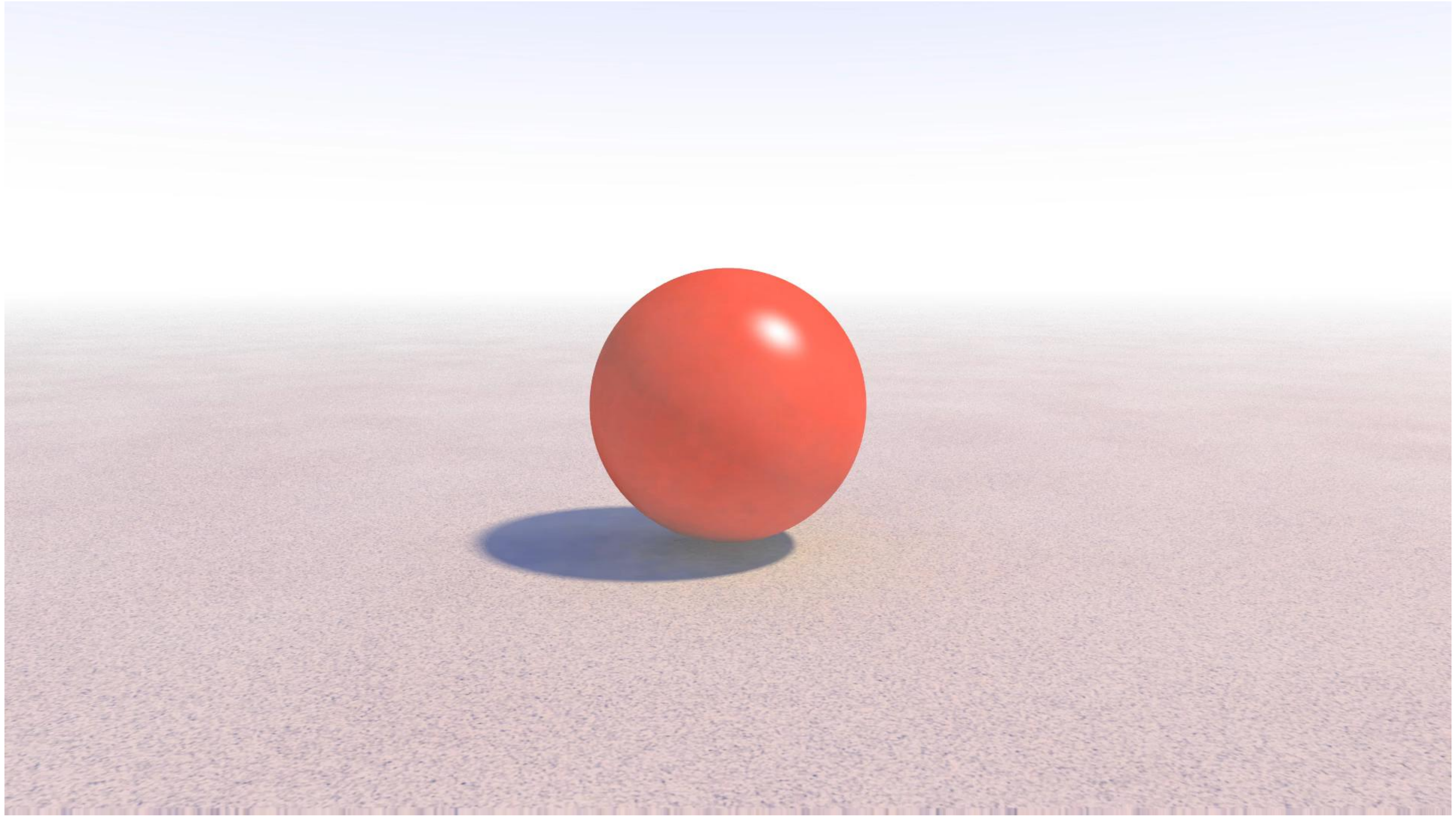
$$F_n = K\delta^n$$

$$K = \frac{4}{3(\sigma_i + \sigma_j)}\left[\frac{R_i R_j}{R_i + R_j}\right]^{0.5}$$

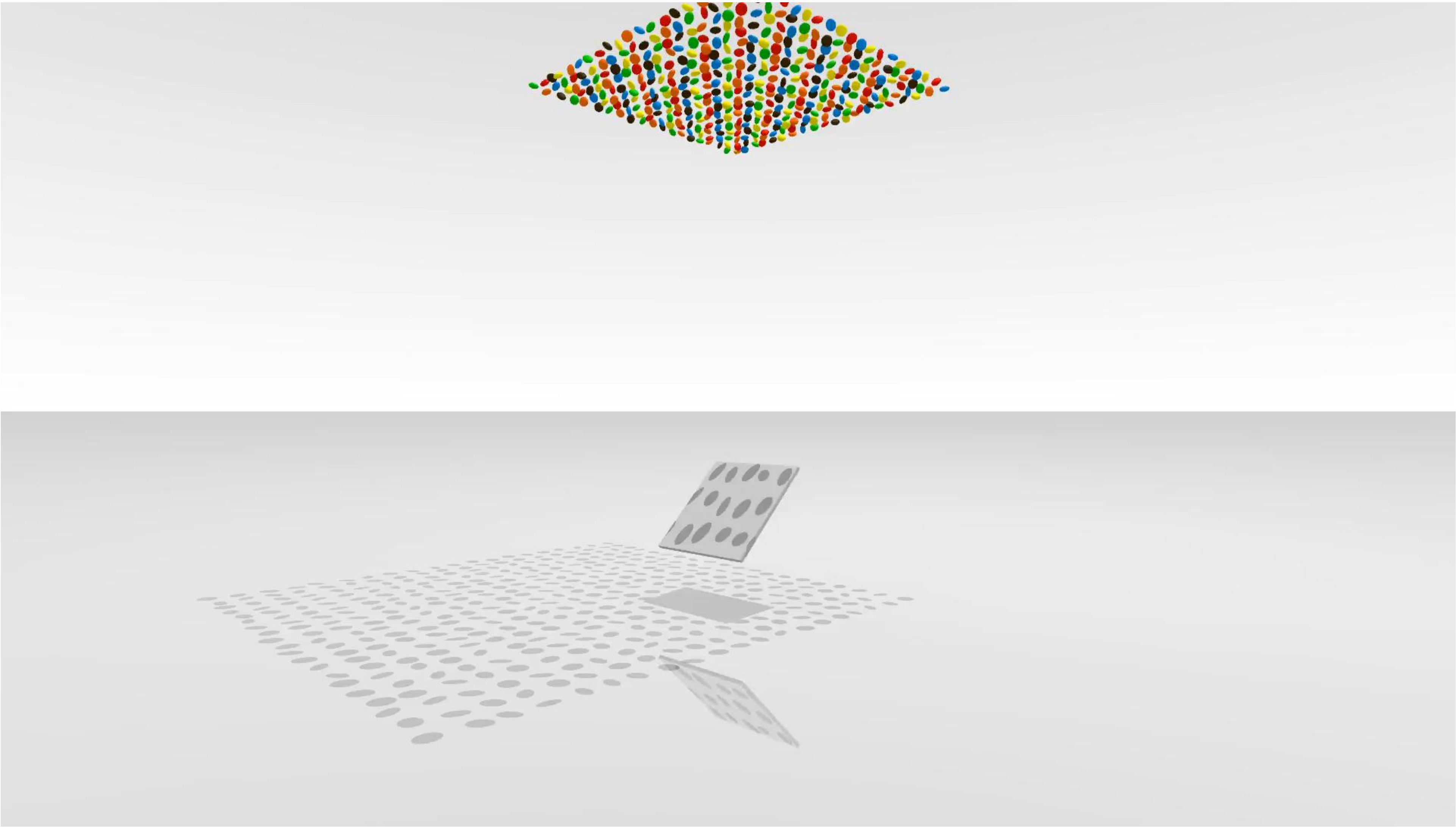


  - Damping and friction can also be introduced
  - Relies on a spherical decomposition of the geometry

Ball – Deformable Net Interaction



Chrono::Rigid - Mixing 50,000 M&Ms on the GPU



Many-Body Dynamics with Friction and Contact

Generalized Positions

Generalized Mass Matrix

Kinematic Differential Equations → $$\dot{\mathbf{q}} = \underbrace{\mathbf{L}(\mathbf{q})}_{\text{Velocity Transformation Matrix}}\underbrace{\mathbf{v}}_{\text{Generalized Velocities}}$$

Force Balance Equations → $$\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} = \underbrace{\mathbf{f}(t,\mathbf{q},\mathbf{v})}_{\text{Applied Force}} - \overbrace{\mathbf{g}_{\mathbf{q}}^{\mathrm{T}}(\mathbf{q},t)\boldsymbol{\lambda}}^{\text{Reaction Force}} + \overbrace{\sum_{i=1}^{N_c}\left(\gamma_n^i \mathbf{D}_n^{T,i} + \gamma_u^i \mathbf{D}_u^{T,i} + \gamma_w^i \mathbf{D}_w^{T,i}\right)}^{\text{Frictional Contact Force}}$$

Holonomic Kinematic Constraints → $$\mathbf{g}(\mathbf{q},t) = \mathbf{0}$$

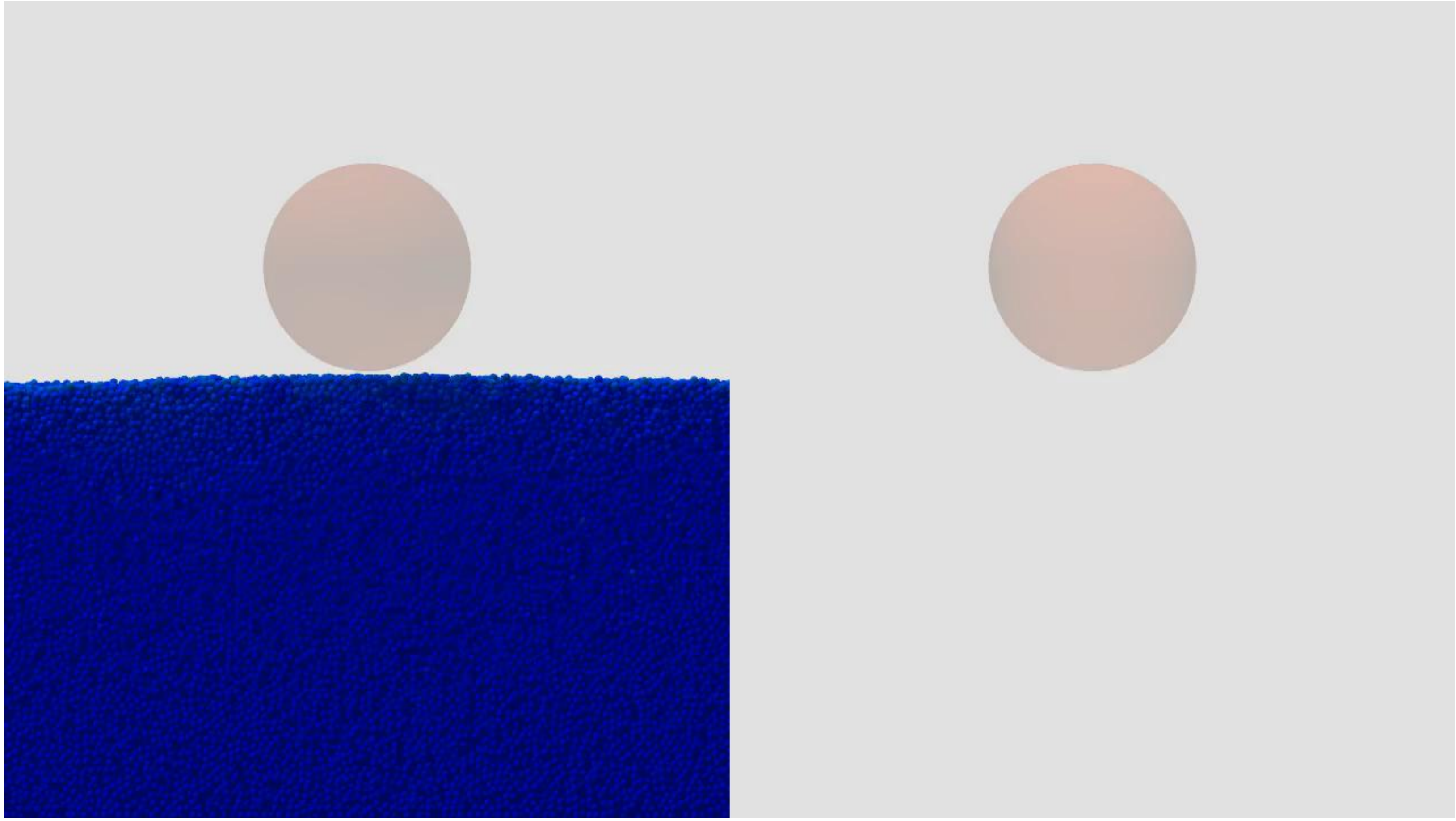
Contact Complementarity Conditions → $$0 \le \Phi^i(\mathbf{q},t) \quad \perp \quad \overbrace{\gamma_n^i}^{\text{Contact Impulse, for Contact ``}i\text{''}} \ge 0 \qquad i = 1,2,\dots,\underbrace{N_c}_{\text{Total Number of Contacts}}$$

Coulomb Friction Model → $$(\gamma_u^i,\gamma_w^i) = \underset{\mu^i\gamma_n^i \ge \sqrt{(\gamma_u^i)^2+(\gamma_w^i)^2}}{\arg\min} \underbrace{\left(\gamma_u^i \mathbf{v}^T \mathbf{D}_u^i + \gamma_w^i \mathbf{v}^T \mathbf{D}_w^i\right)}_{\text{Friction Dissipation Energy}}$$

Gap Function, for Contact "*i*"

Friction Impulse Components, for Contact "*i*"

h=10 cm, ρ_b=2.2 g/cm^3



Chrono::Flow Particle in Suspension Flow

Coupled Problem: Fluid-Solid Interaction

Generalized Positions

Velocity Transformation Matrix

Generalized Mass Matrix

Generalized Velocities

Kinematic Differential Equations → $\dot{\mathbf{q}} = \mathbf{L}(\mathbf{q})\mathbf{v}$

Force Balance Equations → $\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} = \underbrace{\mathbf{f}(t,\mathbf{q},\mathbf{v})}_{\text{Applied Force}} - \overbrace{\mathbf{g}_{\mathbf{q}}^{\mathrm{T}}(\mathbf{q},t)\boldsymbol{\lambda}}^{\text{Reaction Force}} + \overbrace{\sum_{i=1}^{N_c}(\gamma_n^i \mathbf{D}_n^{T,i} + \gamma_u^i \mathbf{D}_u^{T,i} + \gamma_w^i \mathbf{D}_w^{T,i})}^{\text{Frictional Contact Force}}$

Holonomic Kinematic Constraints → $\mathbf{g}(\mathbf{q},t) = \mathbf{0}$

Conservation of mass: $\frac{d\rho}{dt} = -\rho \frac{\partial v^\beta}{\partial x^\beta}$

Conservation of momentum: $\frac{dv^\alpha}{dt} = \frac{1}{\rho}\frac{\partial \sigma^{\alpha\beta}}{\partial x^\beta} + \frac{f^\alpha}{\rho}$

Conservation of energy: $\frac{du}{dt} = \frac{\sigma^{\alpha\beta}}{\rho}\frac{\partial v^\alpha}{\partial x^\beta}$

Algorithmic (applied math) support

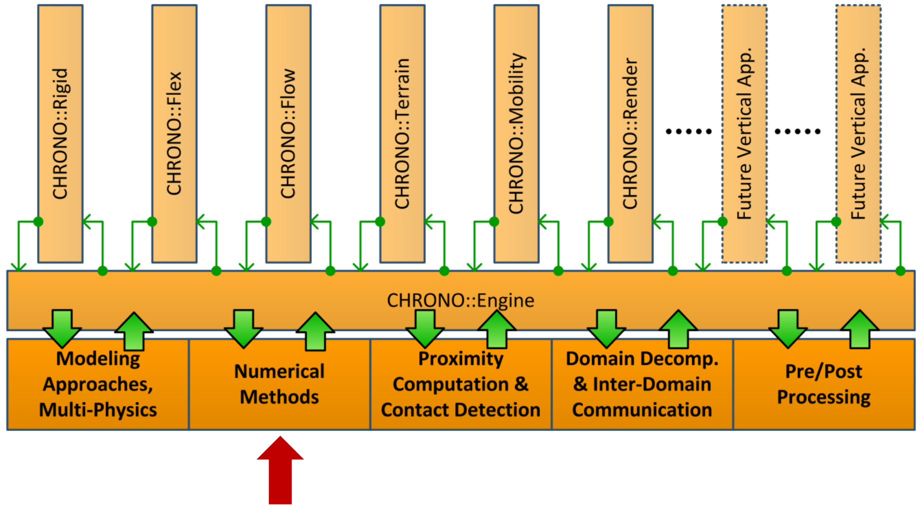
Advanced modeling techniques

Algorithmic (applied math) support

Proximity computation

Domain decomposition & Inter-domain data exchange

Post-processing (visualization)



Deformable Bodies: Implicit Integration using Newmark…

Newmark Integration Formula

- New positions and velocities obtained at t_{n+1} based on new accelerations & Lagrange multipliers

$$\mathbf{q}_{n+1} = \mathbf{q}_n + h\dot{\mathbf{q}}_n + \frac{h^2}{2}\left[(1-2\beta)\ddot{\mathbf{q}}_n + 2\beta\ddot{\mathbf{q}}_{n+1}\right]$$

$$\dot{\mathbf{q}}_{n+1} = \dot{\mathbf{q}}_n + h\left[(1-\gamma)\ddot{\mathbf{q}}_n + \gamma\ddot{\mathbf{q}}_{n+1}\right]$$

Index 3 DAE Problem

- Discretized Equations of Motion at t_{n+1}:

$$(\mathbf{M}\ddot{\mathbf{q}})_{n+1} + (\mathbf{\Phi}_{\mathbf{q}}^T\boldsymbol{\lambda})_{n+1} + (\mathbf{Q}_{\text{int}} - \mathbf{Q}_{\text{ext}})_{n+1} = 0$$

- Kinematic constraints evaluated at new time step t_{n+1}:

$$\mathbf{\Phi}(\mathbf{q}_{n+1}, t_{n+1}) = 0$$

Deformable Bodies: Implicit Integration using Newmark…

- Solving an index 3 set of Differential Algebraic Equations (DAEs) w/ implicit integration
  - Relies on Newton-Krylov approach to solve nonlinear problem at each time step
  - Updates in the accelerations at iteration (k) computed as

$$\begin{bmatrix} \hat{\mathbf{M}} & \mathbf{\Phi}_{\mathbf{q}}^T \\ \mathbf{\Phi}_{\mathbf{q}} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \Delta\ddot{\mathbf{q}} \\ \Delta\boldsymbol{\lambda} \end{bmatrix}^{(k)} = \begin{bmatrix} \text{-}\mathbf{e}_1 \\ \text{-}\mathbf{e}_2 \end{bmatrix}^{(k)}$$

    - Residuals capture error in satisfying the equations of motion and the kinematic constraint equations:

$$\mathbf{e}_1 = \left(\mathbf{M}\ddot{\mathbf{q}}\right)_{n+1} + (\mathbf{\Phi}_{\mathbf{q}}^T\boldsymbol{\lambda})_{n+1} + (\mathbf{Q}_{\text{int}})_{n+1} - (\mathbf{Q}_{\text{ext}})_{n+1}$$

$$\mathbf{e}_2 = \frac{1}{\beta h^2}\mathbf{\Phi}(\mathbf{q}_{n+1}, t_{n+1})$$

Jacobian Matrix Computation, Flex Body Dynamics

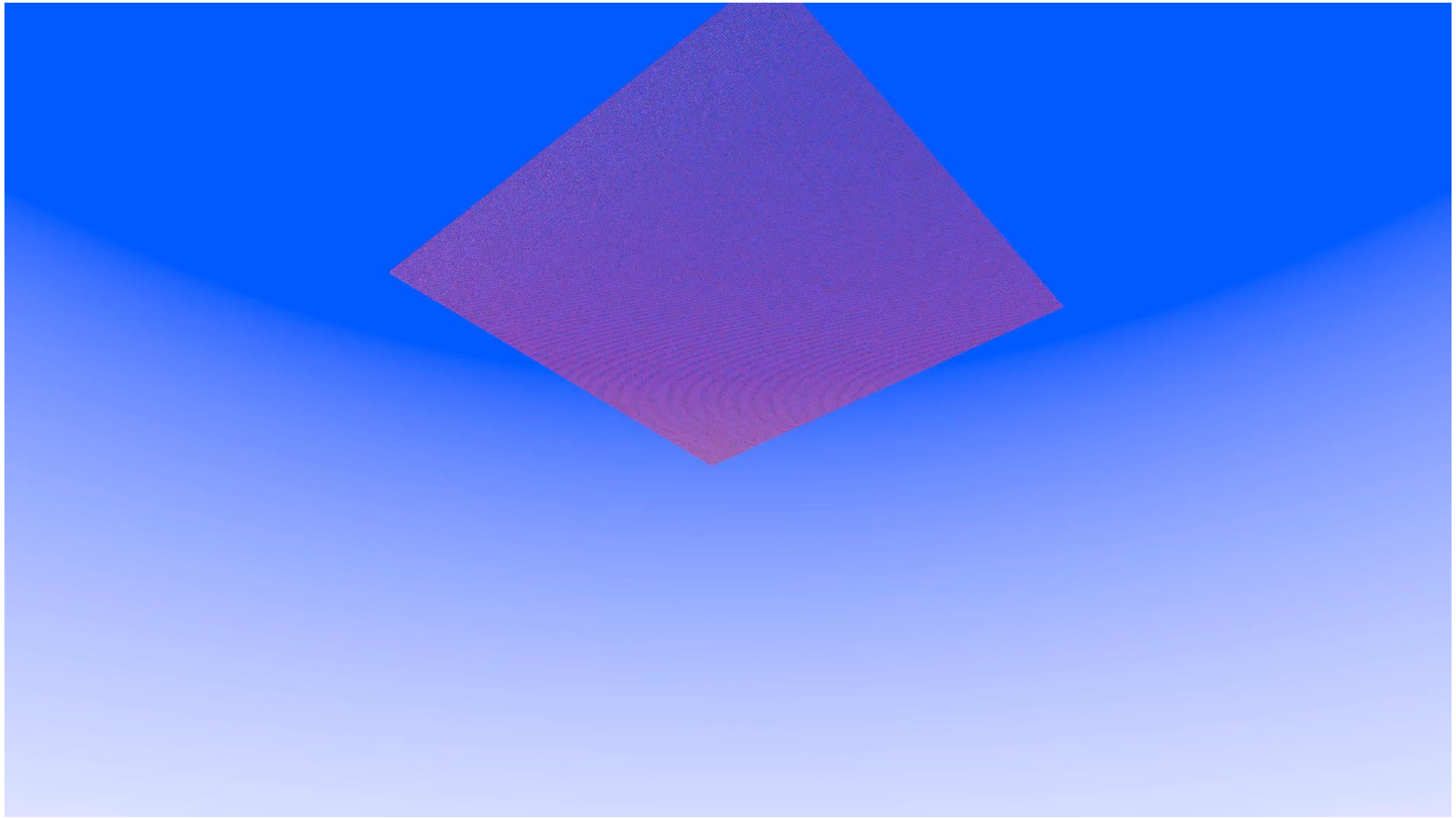
- Sensitivity computation costly

$$\hat{\mathbf{M}} = \frac{\partial \mathbf{e}_1}{\partial \ddot{\mathbf{q}}} = \mathbf{M} - h\gamma \left[\frac{\partial \mathbf{Q}_{ext}}{\partial \dot{\mathbf{q}}} \right] + \beta h^2 \left[(\mathbf{\Phi}_{\mathbf{q}}^T \boldsymbol{\lambda})_{\mathbf{q}} + \frac{\partial \mathbf{Q}_{\text{int}}}{\partial \mathbf{q}} - \frac{\partial \mathbf{Q}_{ext}}{\partial \mathbf{q}} \right]$$

- Computational bottleneck is evaluation of sensitivity of internal forces:

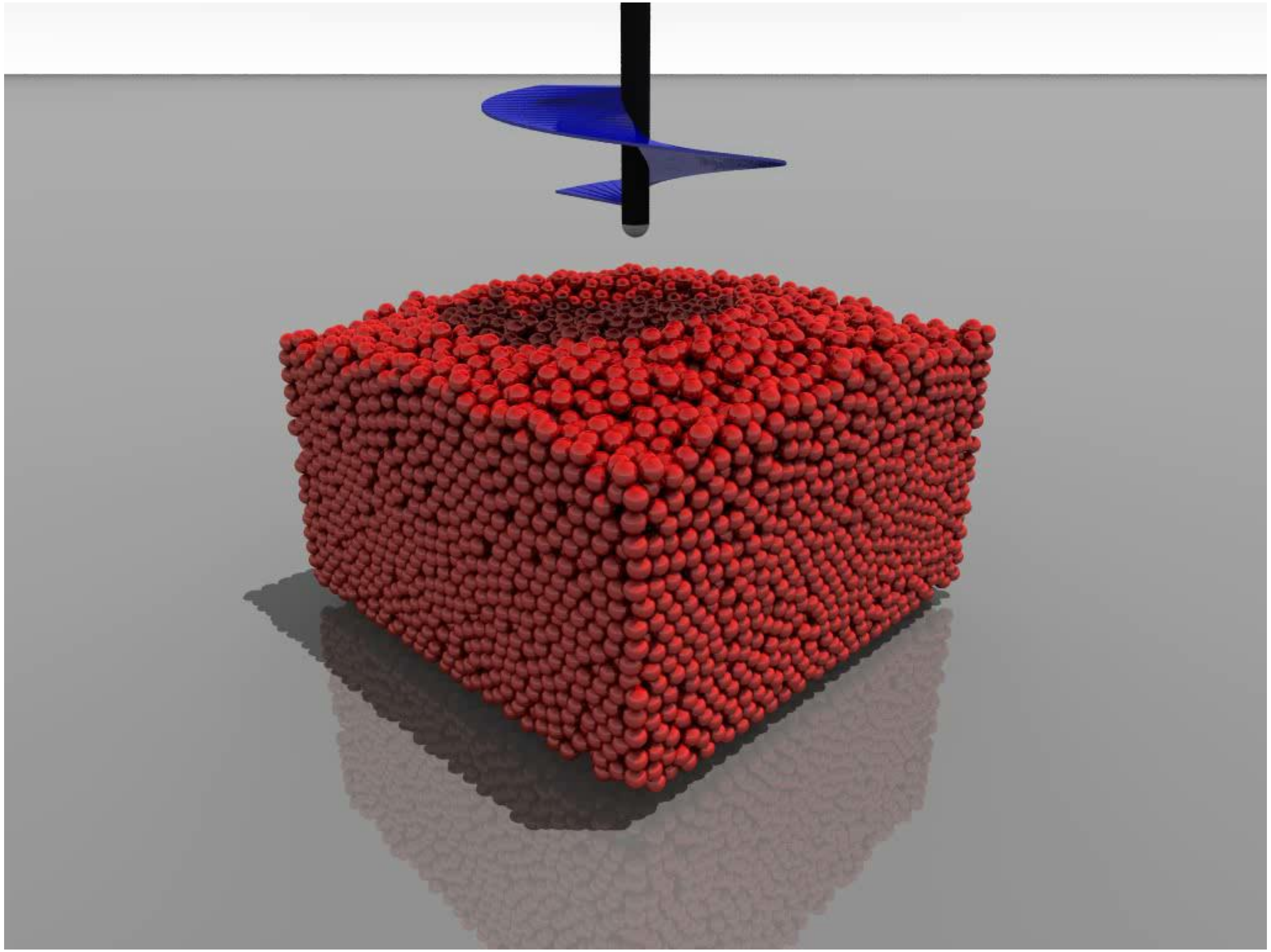
$$\frac{\partial \mathbf{Q}_{\text{int}}}{\partial \mathbf{q}} = \int_0^l EA(\varepsilon_{11}) \frac{\partial}{\partial \mathbf{e}} \left(\frac{\partial \varepsilon_{11}}{\partial \mathbf{e}} \right)^T dx + \int_0^l EA \left(\frac{\partial \varepsilon_{11}}{\partial \mathbf{e}} \right)^T \left(\frac{\partial \varepsilon_{11}}{\partial \mathbf{e}} \right) dx + \int_0^l EI(\kappa) \frac{\partial}{\partial \mathbf{e}} \left(\frac{\partial \kappa}{\partial \mathbf{e}} \right)^T dx + \int_0^l EI \left(\frac{\partial \kappa}{\partial \mathbf{e}} \right)^T \left(\frac{\partial \kappa}{\partial \mathbf{e}} \right) dx$$

0.25 km Net Simulation: 101,025 Beams & 640,146 Constraints

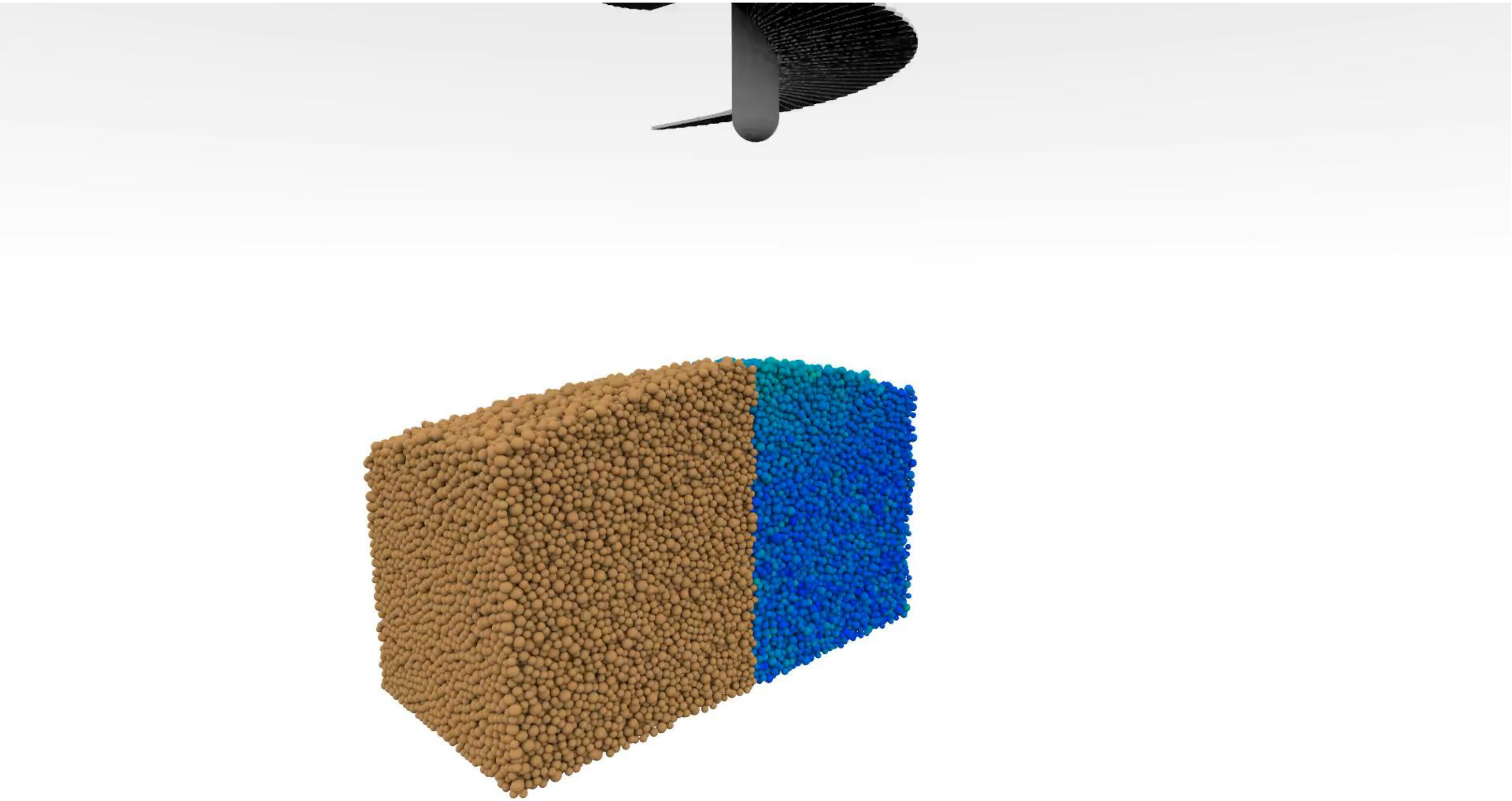


200,000 Bodies & 10 kg Anchor

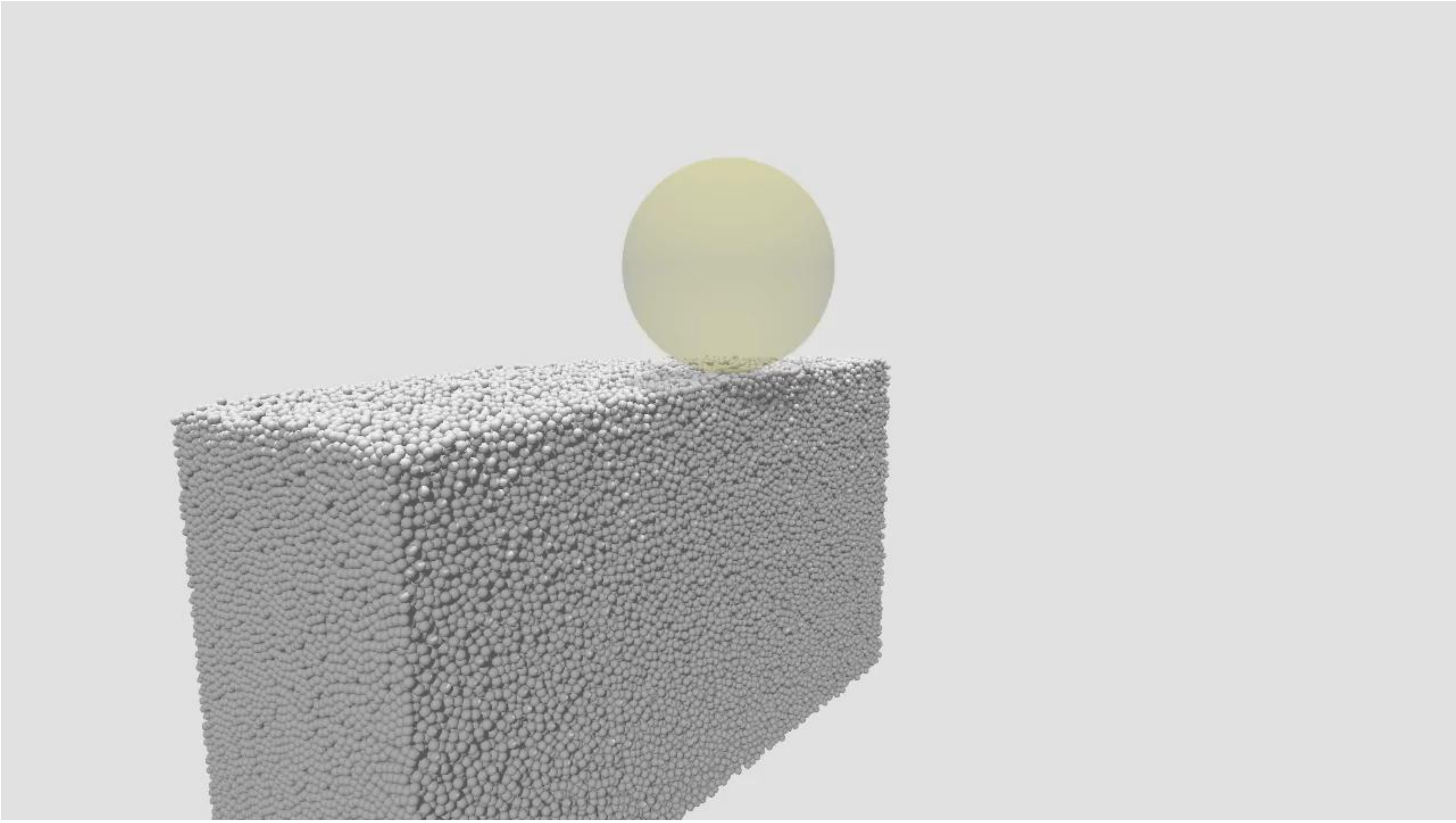
[solved in parallel on the GPU]



Cut-away, Anchoring Simulation

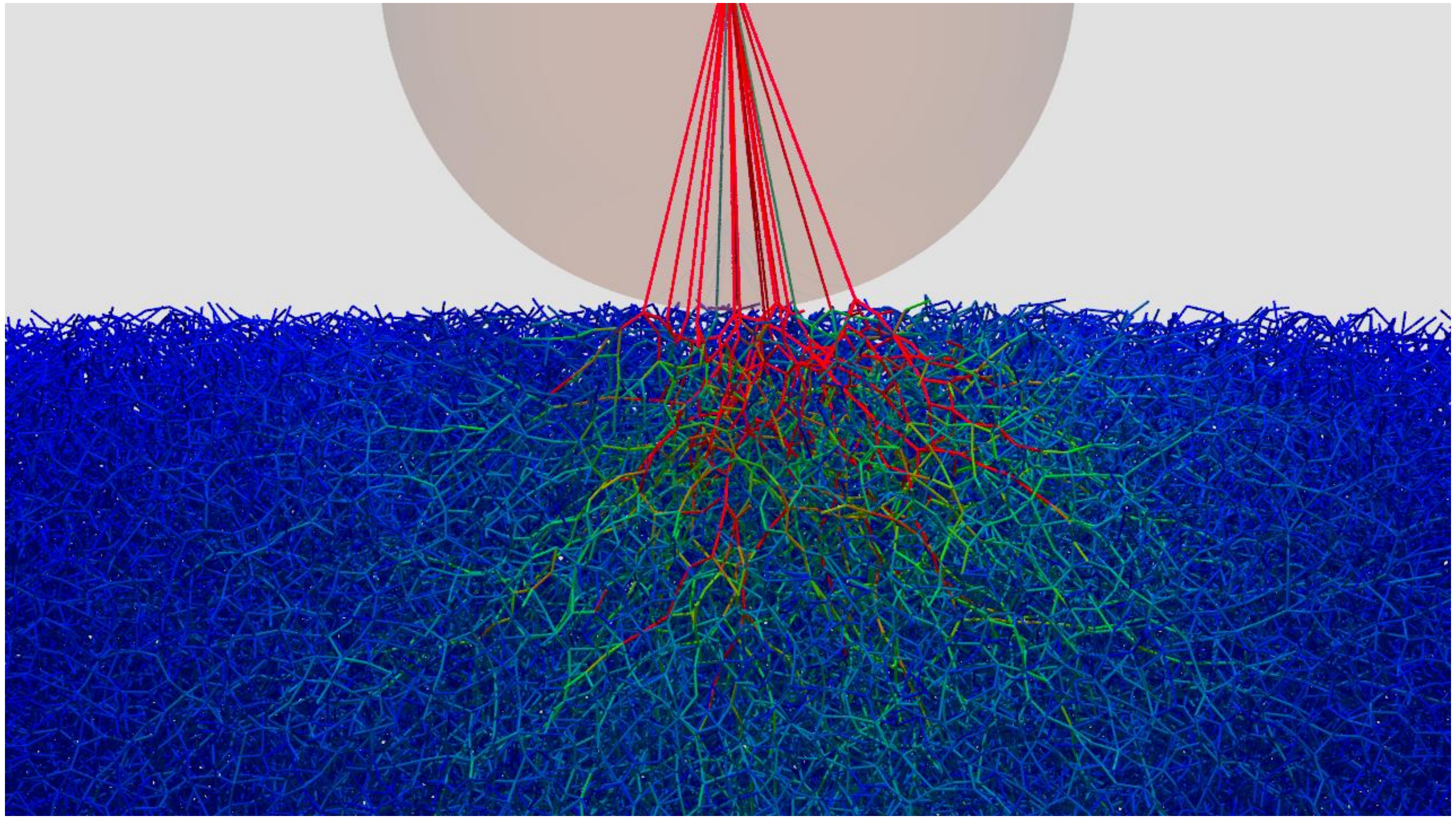


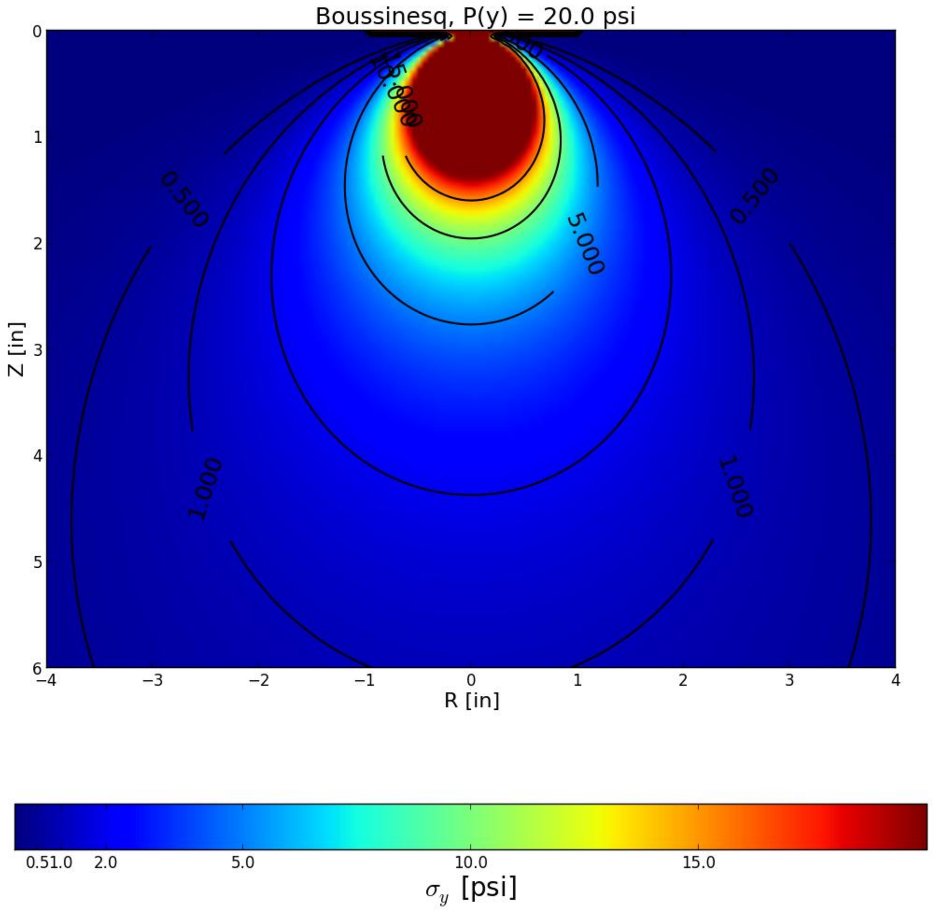
0.5 Million Rigid Bodies



0.5 Million Rigid Bodies

- Granular material has unique properties





UNCLASSIFIED: Distribution Statement A. Approved for public release. #24389

Rigid Body Dynamics w/ DVI: From Continuum to Discrete

- Seeking to solve numerically the equations of motion robustly & effectively
- Discussion focused here on many-body dynamics

Generalized Positions

Velocity Transformation Matrix

Generalized Mass Matrix

Generalized Velocities

Kinematic Differential Equations → $\dot{\mathbf{q}} = \mathbf{T}(\mathbf{q})\mathbf{v}$

Force Balance Equations → $\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} = \underbrace{\mathbf{f}(t,\mathbf{q},\mathbf{v})}_{\text{Applied Force}} - \overbrace{\mathbf{g}_{\mathbf{q}}^{\mathrm{T}}(\mathbf{q},t)\boldsymbol{\lambda}}^{\text{Reaction Force}} + \overbrace{\sum_{i=1}^{N_c}(\gamma_n^i \mathbf{D}_n^{T,i} + \gamma_u^i \mathbf{D}_u^{T,i} + \gamma_w^i \mathbf{D}_w^{T,i})}^{\text{Frictional Contact Force}}$

Holonomic Kinematic Constraints → $\mathbf{g}(\mathbf{q},t) = \mathbf{0}$

Contact Complementarity Conditions → $0 \le \Phi^i(\mathbf{q},t) \perp \overbrace{\gamma_n^i}^{\text{Contact Impulse, for Contact "}i\text{"}} \ge 0 \qquad i = 1,2,\ldots,\underbrace{N_c}_{\text{Total Number of Contacts}}$

Coulomb Friction Model → $(\gamma_u^i, \gamma_w^i) = \underset{\mu^i \gamma_n^i \ge \sqrt{(\gamma_u^i)^2 + (\gamma_w^i)^2}}{\arg\min} \underbrace{(\gamma_u^i \mathbf{v}^T \mathbf{D}_u^i + \gamma_w^i \mathbf{v}^T \mathbf{D}_w^i)}_{\text{Friction Dissipation Energy}}$

Gap Function, for Contact "i"

Friction Impulse Components, for Contact "i"

The Discretized Problem: from t_l to t_{l+1}

positions

time step index

$$\mathbf{q}^{(l+1)} = \mathbf{q}^{(l)} + h\mathbf{L}(\mathbf{q}^{(l)})\mathbf{v}^{(l+1)}$$

Mass Mat.

speeds

Applied Forces

Reaction impulses

$$\mathbf{M}(\mathbf{v}^{(l+1)} - \mathbf{v}^{l}) = h\mathbf{f}(t^{(l)}, \mathbf{q}^{(l)}, \mathbf{v}^{(l)}) + \sum_{i \in \mathcal{A}(q^{(l)}, \delta)} (\gamma_{i,n}\,\mathbf{D}_{i,n} + \gamma_{i,u}\,\mathbf{D}_{i,u} + \gamma_{i,w}\,\mathbf{D}_{i,w})$$

$$i \in \mathcal{A}(q^{(l)}, \delta): \quad 0 \le \frac{1}{h}\Phi_i(\mathbf{q}^{(l)}) + \mathbf{D}_{i,n}^T \mathbf{v}^{(l+1)} \perp \gamma_n^i \ge 0,$$

Complementarity Condition

$$(\gamma_{i,u}, \gamma_{i,w}) = \operatorname*{argmin}_{\mu_i \gamma_{i,n} \ge \sqrt{\gamma_{i,u}^2 + \gamma_{i,w}^2}} \mathbf{v}^T (\gamma_{i,u}\,\mathbf{D}_{i,u} + \gamma_{i,w}\,\mathbf{D}_{i,w}).$$

Stabilization term

Coulomb 3D friction model

(D. Stewart, 1998)

The Cone Complementarity Problem (CCP)

- Define the convex hypercone...

$$\Upsilon = \left(\bigoplus_{i \in \mathcal{A}(\mathbf{q}^l, \epsilon)} \mathcal{FC}^i \right)$$

$\mathcal{FC}^i \in \mathbb{R}^3$ represents friction cone associated with i^{th} contact

- ... and its polar hypercone:

$$\Upsilon^\circ = \left(\bigoplus_{i \in \mathcal{A}(\mathbf{q}^l, \epsilon)} \mathcal{FC}^{i\circ} \right)$$

- The problem can be formulated as find γ that solves the following CCP

$$\gamma \in \Upsilon \perp -(\mathbf{N}\gamma + \mathbf{d}) \in \Upsilon^\circ$$

The Quadratic Programming Angle…

- The CCP captures the first-order optimality condition for a quadratic optimization problem with conic constraints:

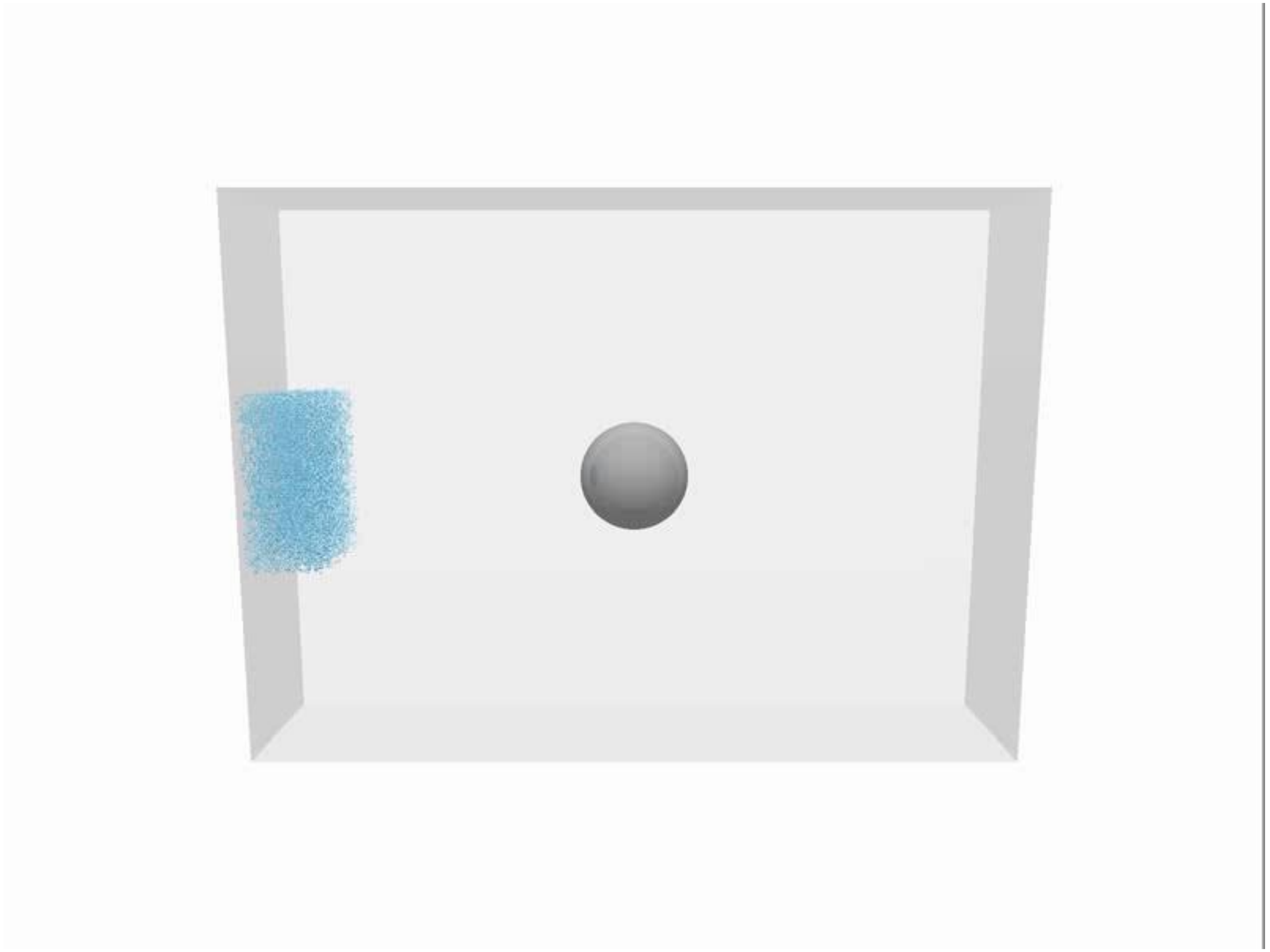
$$\begin{cases} \min \mathbf{q}(\gamma) = \frac{1}{2}\gamma^{\mathbf{T}}\mathbf{N}\gamma + \mathbf{d}^{\mathbf{T}}\gamma \\ \\ \text{subject to} \quad \gamma_i \in \Upsilon_i \;\; \text{for} \;\; i = 1, 2, \ldots, N_c \end{cases}$$

- Notation used:

$$\gamma \equiv [\gamma_1^T, \gamma_2^T, \ldots, \gamma_{N_c}^T]^T \in \mathbb{R}^{3\times N_c} \qquad \text{and} \qquad \Upsilon_i : (\gamma_{u,i}^2 + \gamma_{w,i}^2) - \mu_i^2 \gamma_{n,i}^2 \leq 0$$

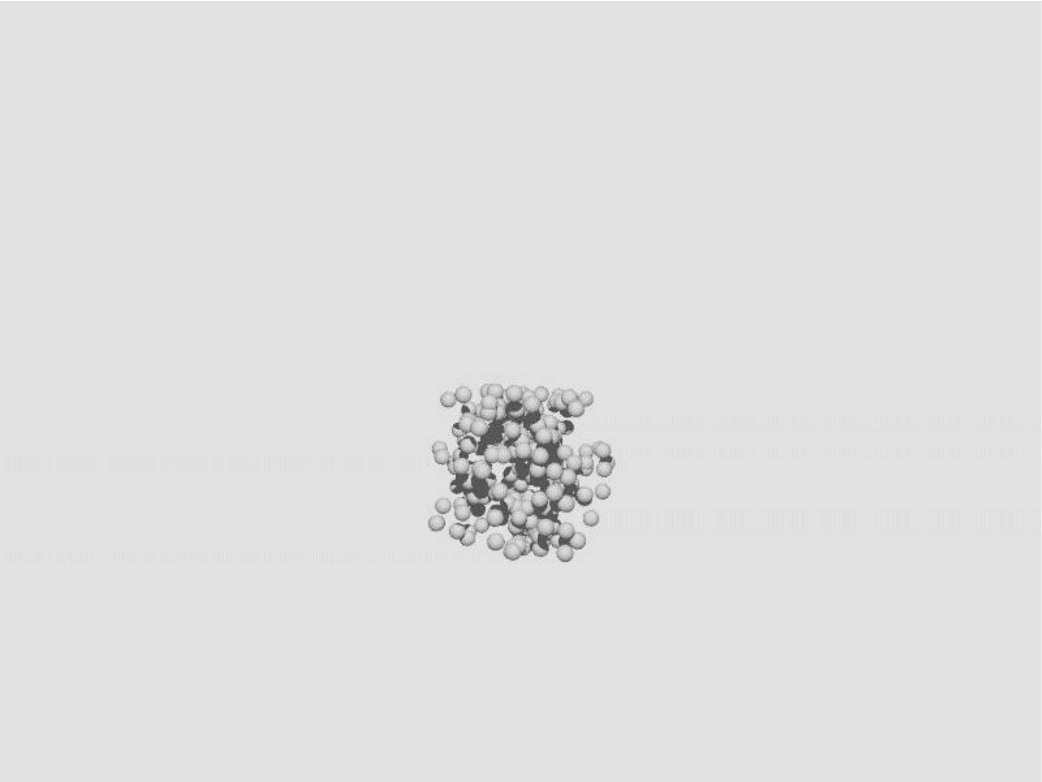
1 Million Rigid Spheres

[parallel on the GPU]

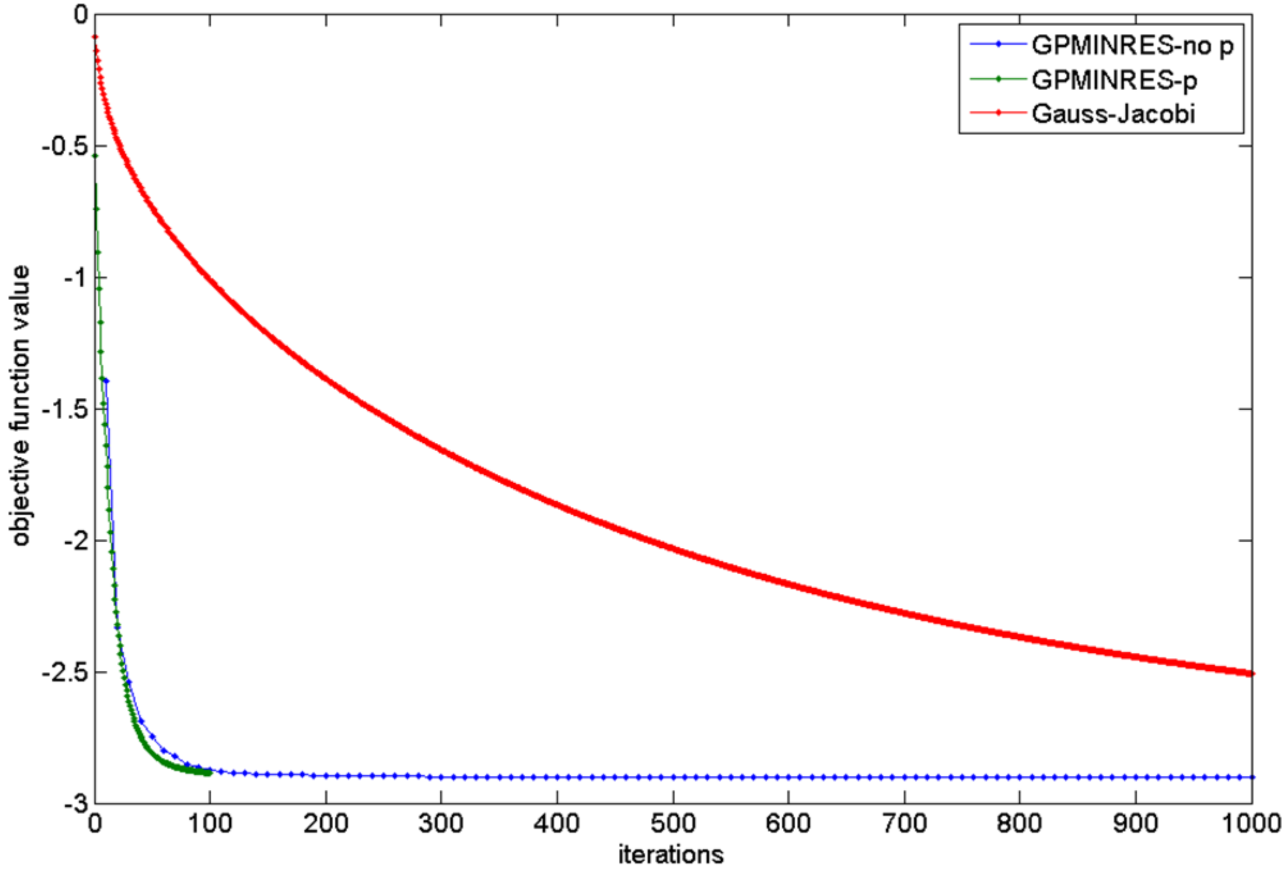


Test Problem: 1000 rigid spheres with 3525 contacts

- Cost function depends on 3,525 variables (or about 10,500 if friction is present)



Objective Function Value [1K bodies, 3525 contacts]



Jacobi

$$
\begin{aligned}
\tilde{\gamma}^{r+1} &= \gamma^r + \omega \mathbf{B}[\mathbf{N}\gamma^r + \mathbf{r}] \\
\hat{\gamma}^{r+1} &= \mathbb{P}(\tilde{\gamma}^{r+1}) \\
\gamma^{r+1} &= \lambda \hat{\gamma}^{r+1} + (1-\lambda)\gamma^r \, .
\end{aligned}
$$

$$
\mathbf{B} = \begin{bmatrix} \eta_1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \eta_{n_c} \end{bmatrix} ,
$$

$$
\eta_i = \frac{1}{\mathrm{Trace}(\mathbf{D}_i^T \mathbf{M}^{-1} \mathbf{D}_i)} .
$$

GP-MINRES

ALGORITHM GPMINRES($\mathbf{N}$, $\mathbf{r}$, τ, η_1, η_2, N_{max}, M_{max})

$\gamma^{(0)} := \mathbf{0}_{nc}$
for $k := 0$ **to** N_{max}
  $\mathbf{y}^{(0)} = \gamma^{(k)}$
  while agressively changing active set and reducing cost function
    $\mathbf{y}^{(j+1)} = P[\mathbf{y}^{(j)} - \alpha_j \nabla q(\mathbf{y}^{(j)})]$
    $j = j + 1$
  endwhile
  $\gamma^{(k)} := \mathbf{y}^{(j)}$
  Determine active set $\mathcal{A}(\gamma^{(k)})$ and $\mathbf{Z}_k$ and $\mathbf{r}_k$
  $\mathbf{w}_0 = \mathbf{0}_{m_k}$
  for $j := 0$ **to** M_{max}
    MINRES step: $\mathbf{w}^{(j)} \to \mathbf{w}^{(j+1)}$
    $j = j + 1$
    if slughish convergence
      break
  enfor
  Set $\bar{\mathbf{w}}_k := \mathbf{w}^{(j)}$
  Get $\gamma^{(k+1)} \to$ backtracking line-search with direction $\mathbf{d}_k = \mathbf{Z}_k \bar{\mathbf{w}}_k$
  if $||\nabla_\Omega q(\gamma^{(k+1)})||_\infty < \tau$
    break
enfor
return Value at time step t_{l+1}, $\gamma^{l+1} := \gamma^{(k+1)}$.

P-SPG-FB

ALGORITHM P-SPG-FB($\mathbf{N}$, $\mathbf{r}$, $\mathbf{x}_0$, $\mathcal{K}$, $\mathbf{P} \mapsto \mathbf{x}$)

$\mathbf{x}_0 := \Pi_\mathcal{K}(\mathbf{x_0})$, $\mathbf{x}_{FB} = \mathbf{x}_0$, $\hat{\alpha}_0 \in [\alpha_{min}, \alpha_{max}]$

$\mathbf{g}_0 := \mathbf{N}\mathbf{x}_0 + \mathbf{r}$, $f(\mathbf{x}_0) = \frac{1}{2}\mathbf{x}_0^T\mathbf{N}\mathbf{x}_0 + \mathbf{x}_0^T\mathbf{r}$, $w_0 = 10^{29}$

for $j := 0$ **to** N_{max}

$\quad \mathbf{p}_j = \mathbf{P}^{-1}\mathbf{g_j}$

$\quad \mathbf{d}_j = \Pi_\mathcal{K}(\mathbf{x}_j - \hat{\alpha}_j\mathbf{p}_j) - \mathbf{x}_j$

$\quad$ **if** $\langle \mathbf{d}_j, \mathbf{g}_j \rangle \geq 0$

$\qquad \mathbf{d}_j = \Pi_\mathcal{K}(\mathbf{x}_j - \hat{\alpha}_j\mathbf{g}_j) - \mathbf{x}_j$

$\quad \lambda := 1$

$\quad$ **while** line search

$\qquad \mathbf{x}_{j+1} := \mathbf{x}_j + \lambda\mathbf{d}_j$

$\qquad \mathbf{g}_{j+1} := \mathbf{N}\mathbf{x}_{j+1} + \mathbf{r}$

$\qquad f(\mathbf{x}_{j+1}) = \frac{1}{2}\mathbf{x}_{j+1}^T\mathbf{N}\mathbf{x}_{j+1} + \mathbf{x}_{j+1}^T\mathbf{r}$

$\qquad$ **if** $f(\mathbf{x}_{j+1}) > \max_{i=0,..,\min(j,N_{GLL})} f(\mathbf{x}_{j-i}) + \gamma\lambda\langle \mathbf{d}_j, \mathbf{g}_j \rangle$

$\qquad\quad$ define $\lambda_{\text{new}} \in [\sigma_{\min}\lambda, \sigma_{\max}\lambda]$ and repeat line search

$\qquad$ **else**

$\qquad\quad$ terminate line search

$\quad \mathbf{s}_j = \mathbf{x}_{j+1} - \mathbf{x}_j$

$\quad \mathbf{y}_j = \mathbf{g}_{j+1} - \mathbf{g}_j$

$\quad$ **if** j is odd

$\qquad \hat{\alpha}_{j+1} = \frac{\langle \mathbf{s}_j, \mathbf{P}\mathbf{s_j} \rangle}{\langle \mathbf{s}_j, \mathbf{y}_j \rangle}$

$\quad$ **else**

$\qquad \hat{\alpha}_{j+1} = \frac{\langle \mathbf{s}_j, \mathbf{y}_j \rangle}{\langle \mathbf{y}_j, \mathbf{P}^{-1}\mathbf{y_j} \rangle}$

$\quad \hat{\alpha}_{j+1} = \min(\alpha_{\max}, \max(\alpha_{\min}, \hat{\alpha}_{j+1}))$

$\quad w_{j+1} = ||[\mathbf{x}_{j+1} - \Pi_\mathcal{K}(\mathbf{x}_{j+1} - \tau_g\mathbf{g}_{j+1})]/\tau_g||_2 = ||\epsilon||_2$

$\quad$ **if** $w_{j+1} \leq \min_{k=0,..,j} w_k$

$\qquad \mathbf{x}_{FB} = \mathbf{x}_{j+1}$

return $\mathbf{x}_{FB}$

Kucera Algorithm

ALGORITHM KUCERA($\mathbf{N}$, $\mathbf{r}$, $\mathbf{x}^0$, $\mathcal{K}$, $\Gamma > 0$, $\tilde{\alpha} \in (0, \|\mathbf{N}\|^{-1}]$, $\epsilon > 0$)

$$
\begin{array}{l}
k = 0 \\
\mathbf{g} = \mathbf{N}\mathbf{x}^0 + \mathbf{r} \\
\mathbf{p} = \phi(\mathbf{x}^0) \\
\textbf{while } \|\tilde{\mathbf{g}}(\mathbf{x}^k)\| > \epsilon \\
\quad \textbf{if } \tilde{\beta}(\mathbf{x}^k)^T \mathbf{g}(\mathbf{x}^k) \leq \Gamma^2 \tilde{\phi}(\mathbf{x}^k)^T \mathbf{g}(\mathbf{x}^k) \\
\quad\quad \alpha_{cg} = \mathbf{g}^T \mathbf{p} / \mathbf{p}^T \mathbf{N} \mathbf{p} \\
\quad\quad \alpha_f = min(\alpha_{f,i}) \text{ where } \alpha_{f,i} = \begin{cases} \mathbf{x}_i / \mathbf{p}_i, & \text{if } \mathbf{p}_i > 0 \\ \infty, & \text{if } \mathbf{p}_i \leq 0 \end{cases} \\
\quad\quad \textbf{if } \alpha_{cg} < \alpha_f \\
\quad\quad\quad \mathbf{x}^{k+1} = \mathbf{x}^k - \alpha_{cg} \mathbf{p} \\
\quad\quad\quad \mathbf{g} = \mathbf{g} - \alpha_{cg} \mathbf{N} \mathbf{p} \\
\quad\quad\quad \gamma = \phi(\mathbf{x}^{k+1})^T \mathbf{N} \mathbf{p} / \mathbf{p}^T \mathbf{A} \mathbf{p} \\
\quad\quad\quad \mathbf{p} = \phi(\mathbf{x}^{k+1}) - \gamma \mathbf{p} \\
\quad\quad \textbf{else} \\
\quad\quad\quad \mathbf{x}^{k+1/2} = \mathbf{x}^k - \alpha_f \mathbf{p} \\
\quad\quad\quad \mathbf{x}^{k+1} = \mathbf{x}^{k+1/2} - \tilde{\alpha} \tilde{\phi}(\mathbf{x}^{k+1/2}) \\
\quad\quad\quad \mathbf{g} = \mathbf{N} \mathbf{x}^{k+1} + \mathbf{r} \\
\quad\quad\quad \mathbf{p} = \phi(\mathbf{x}^{k+1}) \\
\quad \textbf{else} \\
\quad\quad \mathbf{x}^{k+1} = \mathbf{x}^k - \tilde{\alpha} \tilde{\beta}(\mathbf{x}^k) \\
\quad\quad \mathbf{g} = \mathbf{N} \mathbf{x}^{k+1} + \mathbf{r} \\
\quad\quad \mathbf{p} = \phi(\mathbf{x}^{k+1}) \\
\quad k = k + 1 \\
\textbf{return } \mathbf{x}^k
\end{array}
$$

Nesterov's Accelerated Projected Gradient Descent

ALGORITHM NAPG($\boldsymbol{N}, \boldsymbol{r}, t \leq \frac{1}{\lambda_{max}(\boldsymbol{N})}, \tau, N_{max}$)

$\boldsymbol{\gamma}_0 = \boldsymbol{0}_{n_c}$

$\hat{\boldsymbol{\gamma}}_0 = \boldsymbol{1}_{n_c}$

$\boldsymbol{y}_0 = \boldsymbol{\gamma}_0$

$\theta_0 = 1$

for $k := 0$ **to** N_{max}

    $\boldsymbol{g} = \boldsymbol{N}\boldsymbol{y}_k - \boldsymbol{r}$

    $\boldsymbol{\gamma}_{k+1} = \Pi_{\mathcal{K}}\left(\boldsymbol{y}_k - t\boldsymbol{g}\right)$

    $\theta_{k+1} = \frac{-\theta_k^2 + \theta_k\sqrt{\theta_k^2+4}}{2}$

    $\beta_{k+1} = \theta_k \frac{1-\theta_k}{\theta_k^2+\theta_{k+1}}$

    $\boldsymbol{y}_{k+1} = \boldsymbol{\gamma}_{k+1} + \beta_{k+1}\left(\boldsymbol{\gamma}_{k+1} - \boldsymbol{\gamma}_k\right)$

    $\epsilon = \epsilon\left(\boldsymbol{\gamma}_{k+1}\right)$

    if $\epsilon < \tau$

        break

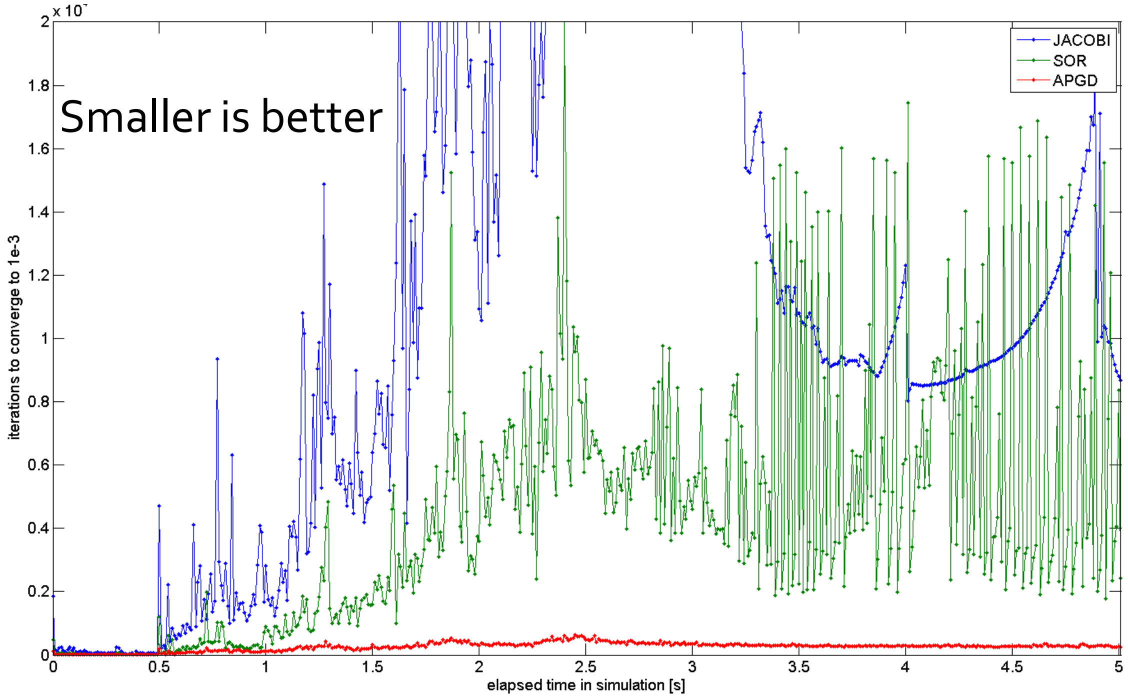
    endif

endfor

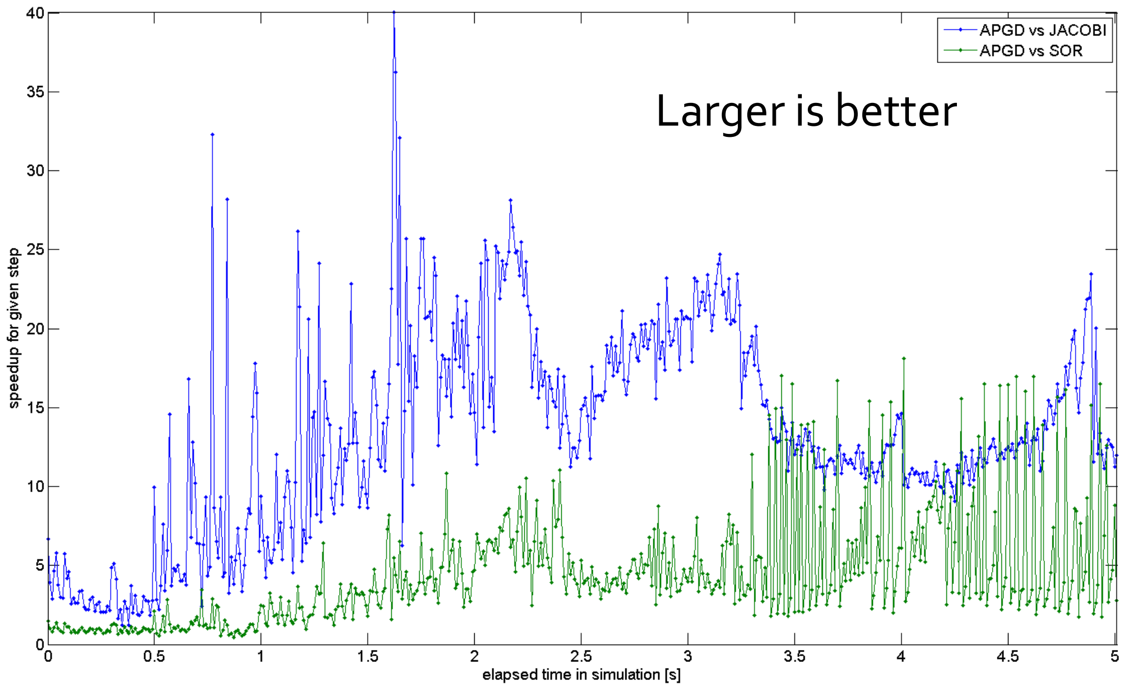
return Value at time step t_{l+1}, $\boldsymbol{\gamma}^{l+1} := \hat{\boldsymbol{\gamma}}$.

Relative Speedup: Benchmark Problem [1000 bodies]

Number of iterations to convergence



Speedup: APGD vs. Jacobi and SOR



Proximity computation

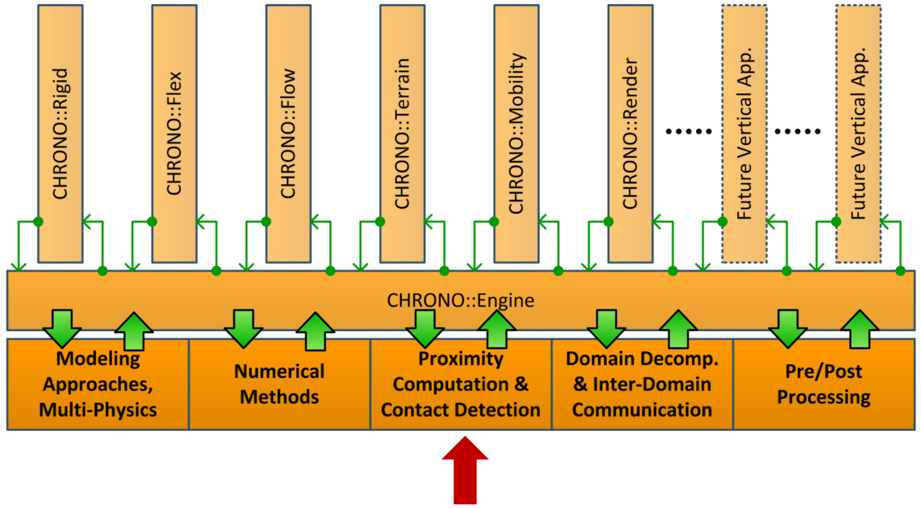
Advanced modeling techniques

Algorithmic (applied math) support

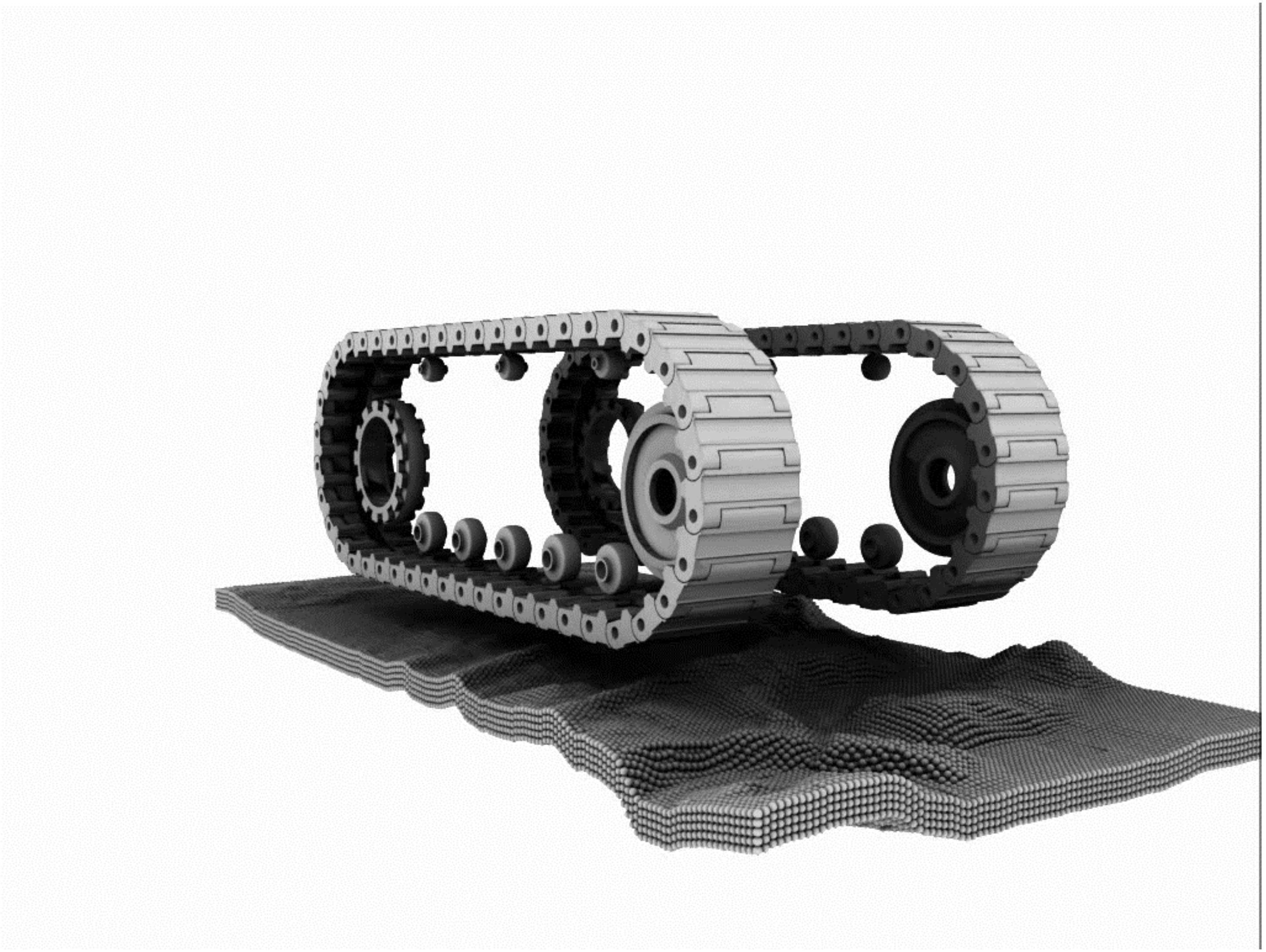
Proximity computation

Domain decomposition & Inter-domain data exchange

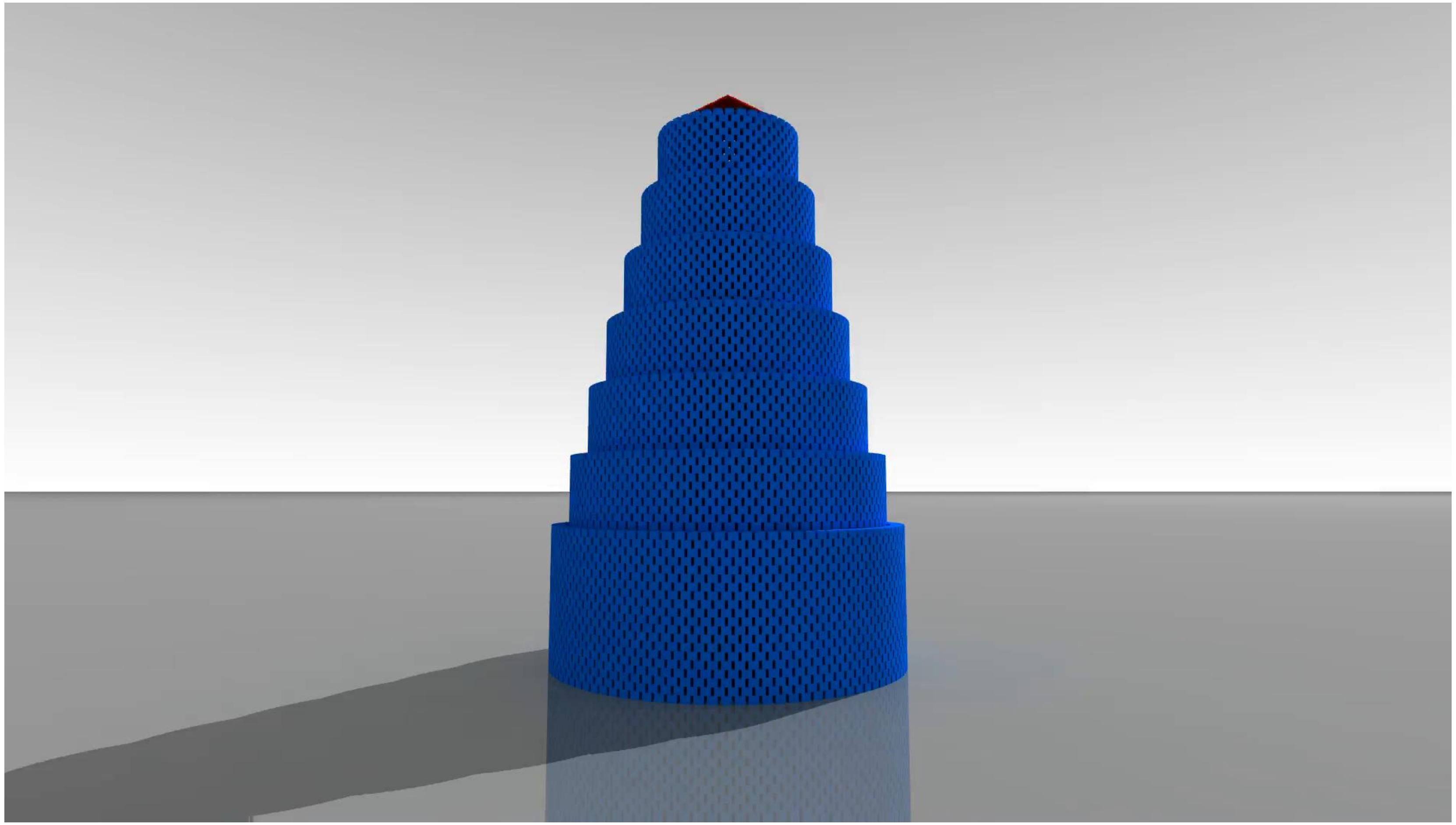
Post-processing (visualization)



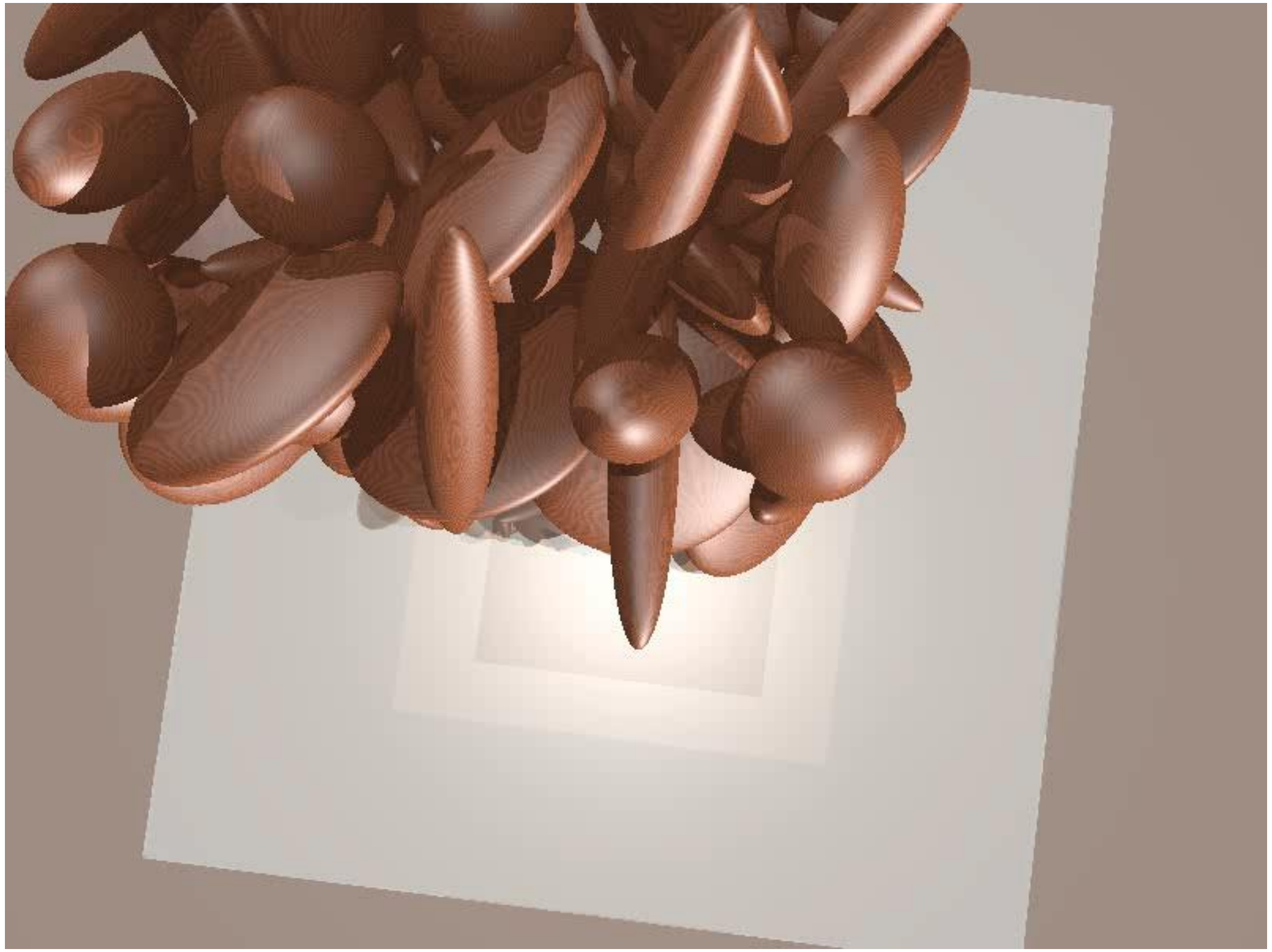
Tracked Vehicle Simulation



600,000 Bodies Moving & Colliding
[run on the GPU]



Ellipsoid-Ellipsoid CD



Various Geometries Handled… [Ellipsoid-Ellipsoid Example]

$$\mathbf{d} = \mathbf{P}_1 - \mathbf{P}_2 = (\frac{1}{2\lambda_1}\mathbf{M}_1 + \frac{1}{2\lambda_2}\mathbf{M}_2)\mathbf{c} + (\mathbf{b}_1 - \mathbf{b}_2)$$

$$\frac{\partial \mathbf{d}}{\partial \alpha_i} = \frac{\partial \mathbf{P}_1}{\partial \alpha_i} - \frac{\partial \mathbf{P}_2}{\partial \alpha_i} \quad , \quad \frac{\partial^2 \mathbf{d}}{\partial \alpha_i \partial \alpha_j} = \frac{\partial^2 \mathbf{P}_1}{\partial \alpha_i \partial \alpha_j} - \frac{\partial^2 \mathbf{P}_2}{\partial \alpha_i \partial \alpha_j}$$

$$\boxed{\frac{\partial \mathbf{P}}{\partial \alpha_i} = (\frac{1}{2\lambda}\mathbf{M} - \frac{1}{8\lambda^3}\mathbf{M}\mathbf{c}\mathbf{c}^T\mathbf{M})\frac{\partial \mathbf{c}}{\partial \alpha_i}}$$

$$\boxed{\begin{aligned}\frac{\partial^2 \mathbf{P}}{\partial \alpha_i \partial \alpha_j} &= (-\frac{1}{8\lambda^3}\mathbf{M} + \frac{3}{32\lambda^5}\mathbf{M}\mathbf{c}\mathbf{c}^T\mathbf{M})\mathbf{c}^T\mathbf{M}\frac{\partial \mathbf{c}}{\partial \alpha_j}\frac{\partial \mathbf{c}}{\partial \alpha_i} \\ &\quad - \frac{1}{8\lambda^3}[(\mathbf{c}^T\mathbf{M}\frac{\partial \mathbf{c}}{\partial \alpha_i})\mathbf{M} + \mathbf{M}\mathbf{c}(\frac{\partial \mathbf{c}}{\partial \alpha_i})^T\mathbf{M}]\frac{\partial \mathbf{c}}{\partial \alpha_j} \\ &\quad + (\frac{1}{2\lambda}\mathbf{M} - \frac{1}{8\lambda^3}\mathbf{M}\mathbf{c}\mathbf{c}^T\mathbf{M})\frac{\partial^2 \mathbf{c}}{\partial \alpha_i \partial \alpha_j}\end{aligned}}$$

$$\varepsilon: \ \frac{x^2}{r_1^2} + \frac{y^2}{r_2^2} + \frac{z^2}{r_3^2} = 1$$

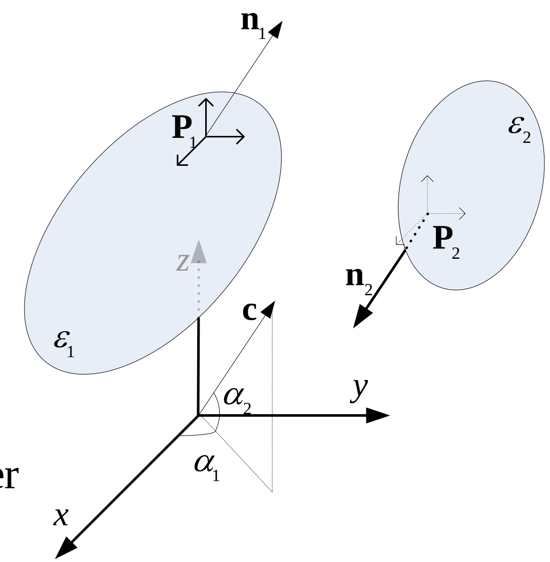
$\mathbf{A}$: Rotation Matrix

$$\mathbf{M} = \mathbf{A}\mathbf{R}^2\mathbf{A}^T$$

$$\mathbf{R} = diag(r_1, r_2, r_3)$$

$\mathbf{b}$: Translation of ellipsoids center

$$\lambda^2 = \frac{1}{4}\mathbf{n}^T\mathbf{M}\mathbf{n}$$



$$\mathbf{d} = \mathbf{P}_1 - \mathbf{P}_2$$

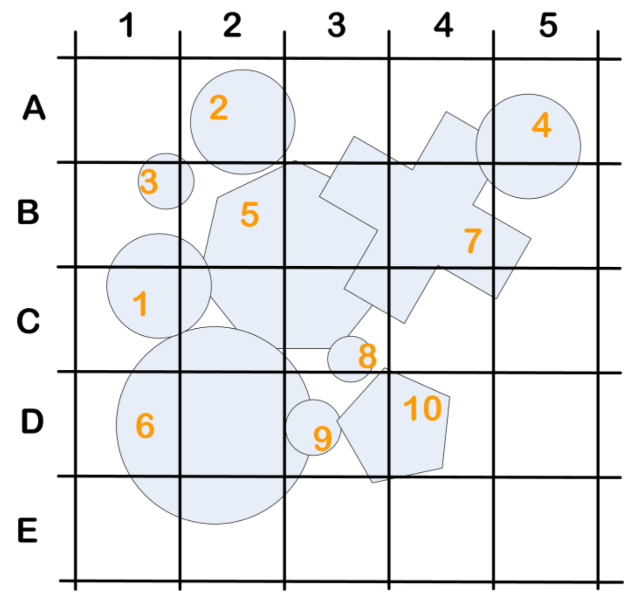
$$\min_{\alpha_1, \alpha_2} \|d(\alpha_1, \alpha_2)\|^2$$

Collision Detection

- Broad phase
  - Draws on an Axis Aligned Bounding Box (AABB) approach
- Narrow phase
  - Draws on Minkowski Portal Refinement

CD: Binning

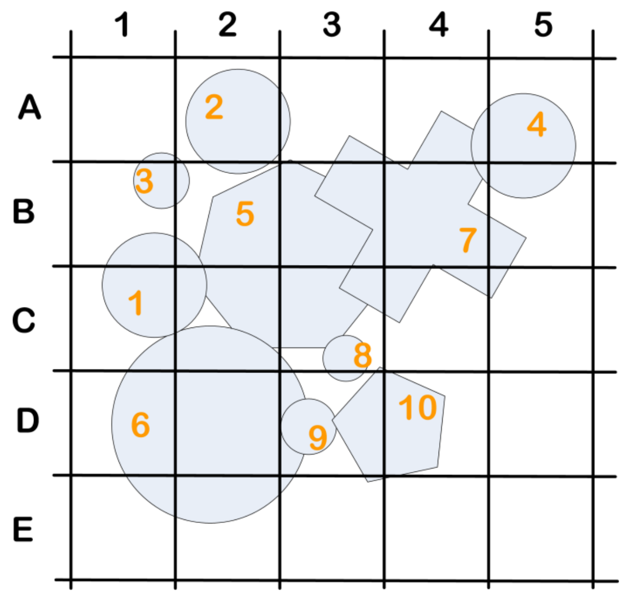
- Example: 2D collision detection, bins are squares

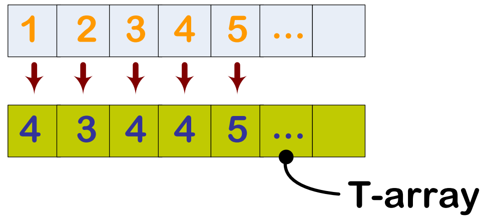


- Body 4 touches bins A4, A5, B4, B5
- Body 7 touches bins A3, A4, A5, B3, B4, B5, C3, C4, C5
- In proposed algorithm, bodies 4 and 7 will be checked for collision by three threads (associated with bin A4, A5, B4)

Stage 1 (Body Parallel)

- Purpose: find the number of bins touched by each body
- Store results in the "**T**", array of N integers
- Key observation: it's easy to bin bodies



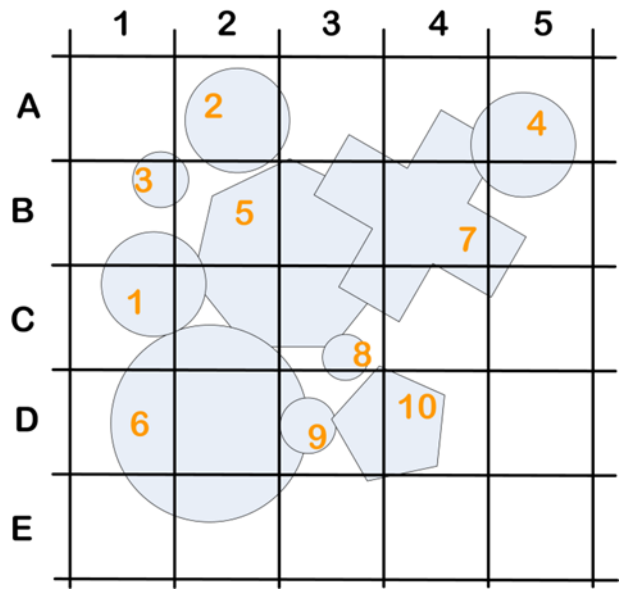


Stage 2: Parallel Inclusive Scan

- Run a parallel inclusive scan on the array **T**
  - The last element is the total number of bin touches, including the last body

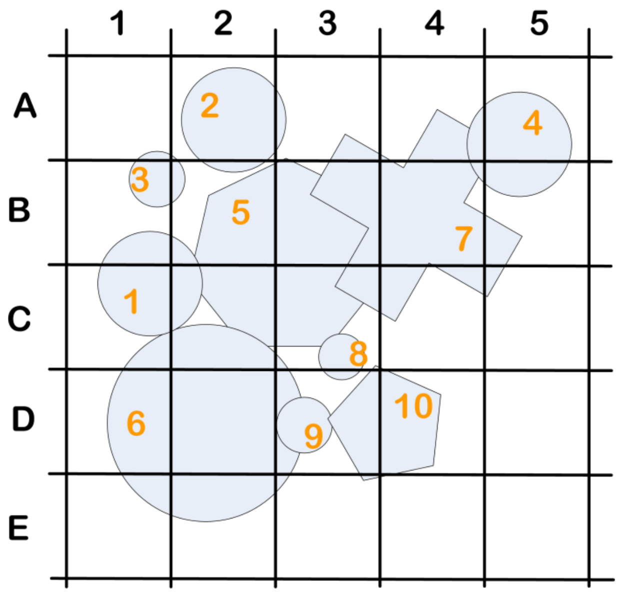
- Complexity of Stage: O(N) – using **thrust** library

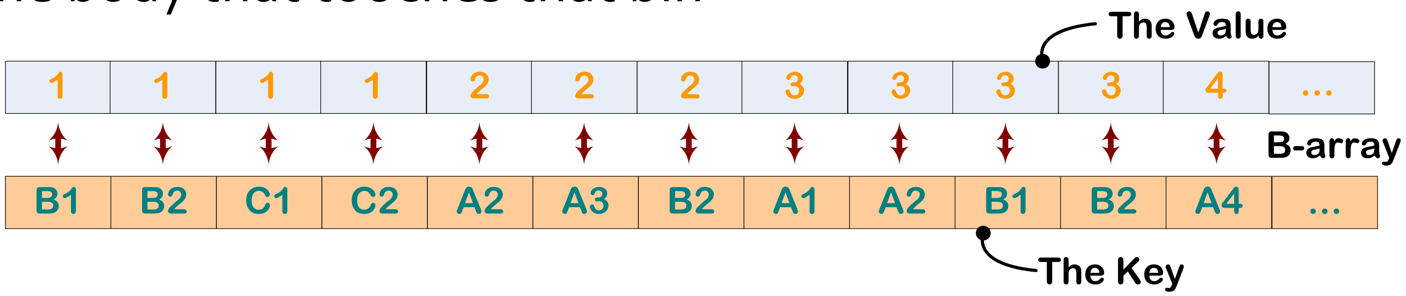
- Purpose: determine the number of entries M needed to store the indices of all the bins touched by each body in the problem



Stage 3: Determine bin-to-body association

- Stage executed in parallel on a per-body basis
- Allocate an array **B** of M pairs of integers.
- The key (first entry of the pair), is the bin index
- The value (second entry of pair) is the body that touches that bin

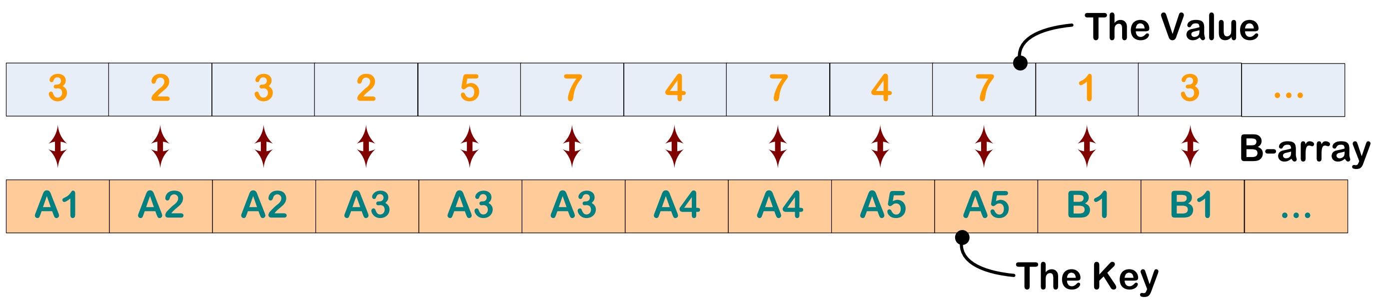


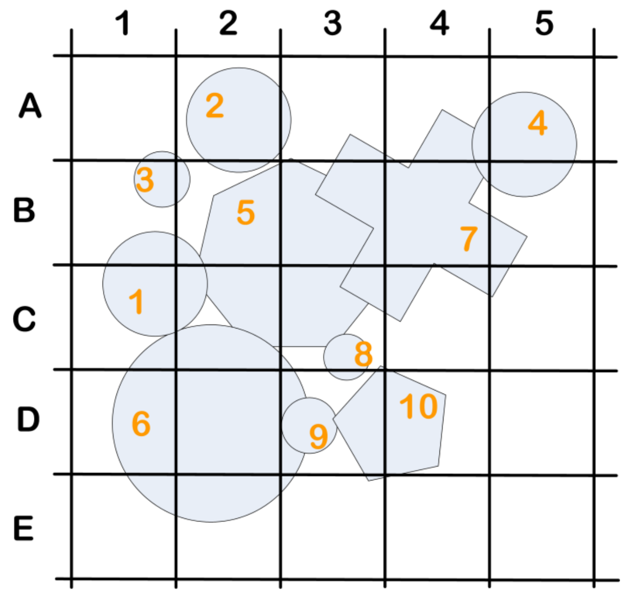


Stage 4: Radix Sort

- In parallel, run radix sort to order the **B** array according to key values

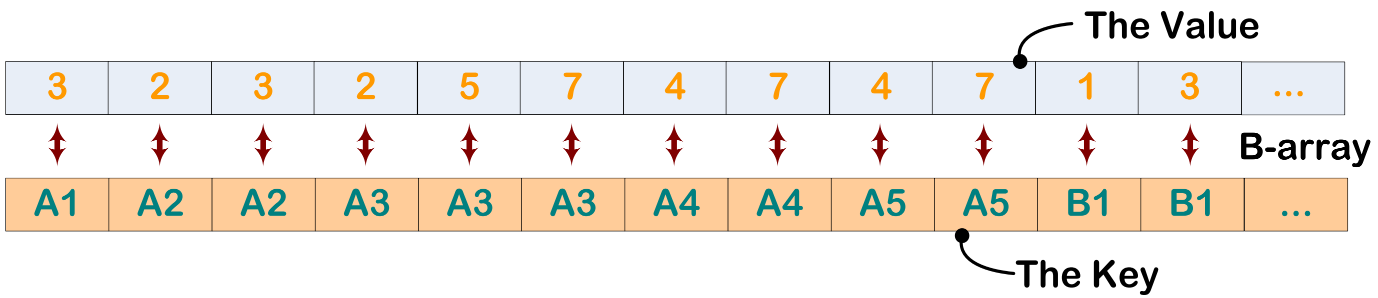
- Work load: O(N)
- Relies on **thrust** library

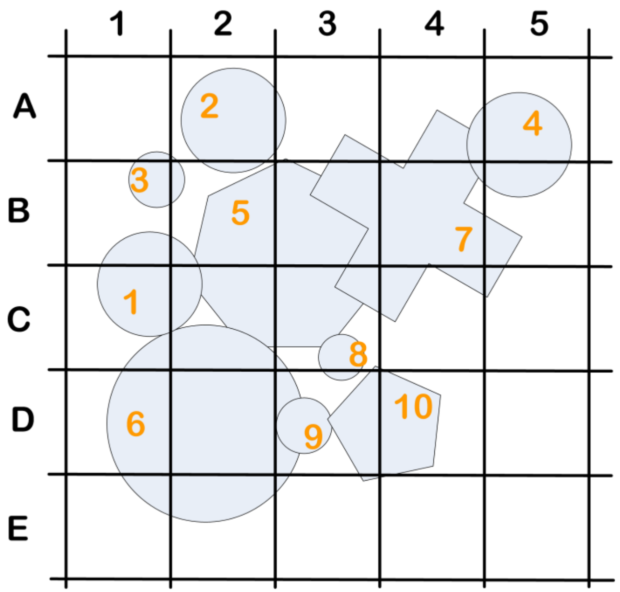




Stage 5: Find Bin Starting Index

- Host allocates on device an array of length N_b of pairs of unsigned integers
- Run in parallel, on a per bin basis:
  - Load in parallel in shared memory chunks of the **B** array and find the location where each bin starts
  - Store it in entry *k* of **C**, as the key associated with this pair
  - Key of bins with one or no bodies is set to maximum unsigned int value of 0xffffffff





Stage 6: Sort **C** for Pruning

- Do a parallel radix sort on the array **C** based on the key
- Purpose: move unused bins to the end of array
- Effort: $O(N_b)$

C-array

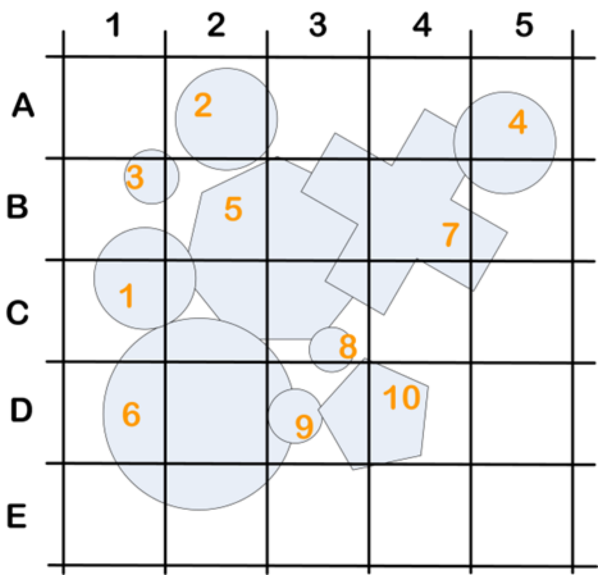
The Value	0	1	3	6	8	10	...
The Key	0xfff	2	3	2	2	2	...
	A1	A2	A3	A4	A5	B1	...

Becomes this

C-array

The Value	1	6	8	10	3	...	
The Key	2	2	2	2	3	...	

...



Stage 7: Investigate Collisions in each Bin

- Carried out in parallel, one thread per bin

B-array

0	1	2	3	4	5	6	7	8	9	10	11	
3	2	3	2	5	7	4	7	4	7	1	3	...
A1	A2	A2	A3	A3	A3	A4	A4	A5	A5	B1	B1	...

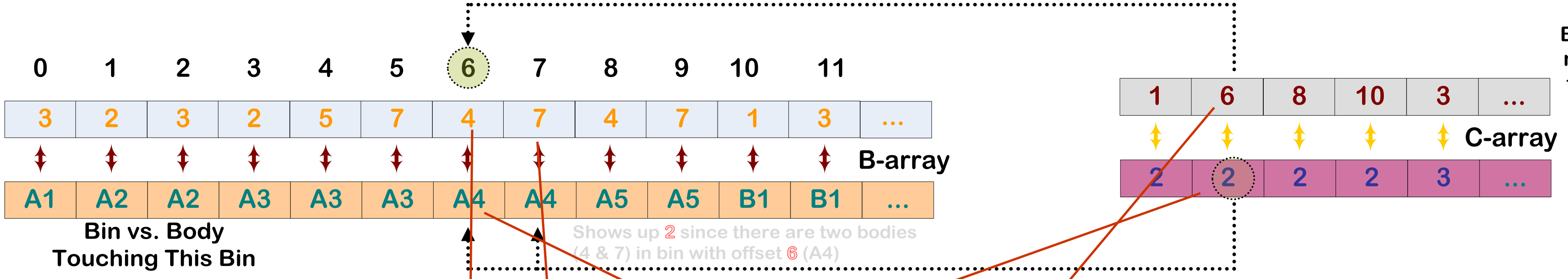
C-array

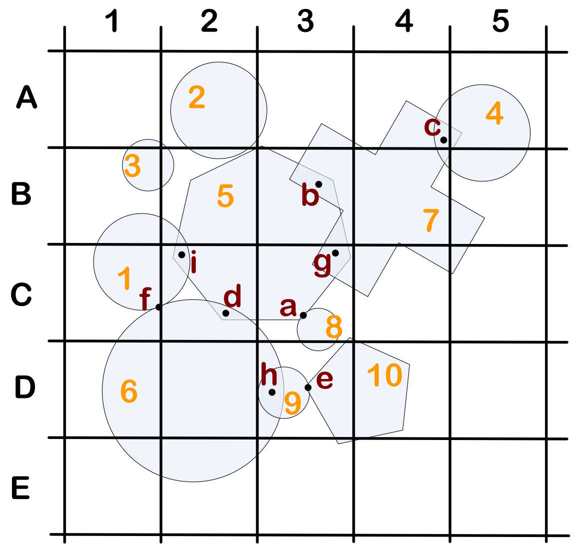
1	6	8	10	3	...
2	2	2	2	3	...

- To store information generated during this stage host allocates unsigned integer array **D** of length N_b
  - Array **D** stores the number of actual contacts occurring in each bin
  - **D** is in sync with (linked to) **C**, which in turn is in sync with (linked to) **B**
- Parallelism: one thread per bin
  - Thread *k* reads the pair key-value in entry *k* of array **C**
  - Thread *k* reads does rehearsal for brute force collision detection
  - Outcome: the number s of active collisions taking place in a bin
  - Value *s* stored in k^{th} entry of the **D** array

Stage 7: Details…

- Recall that how C is organized is a reflection of how B is organized





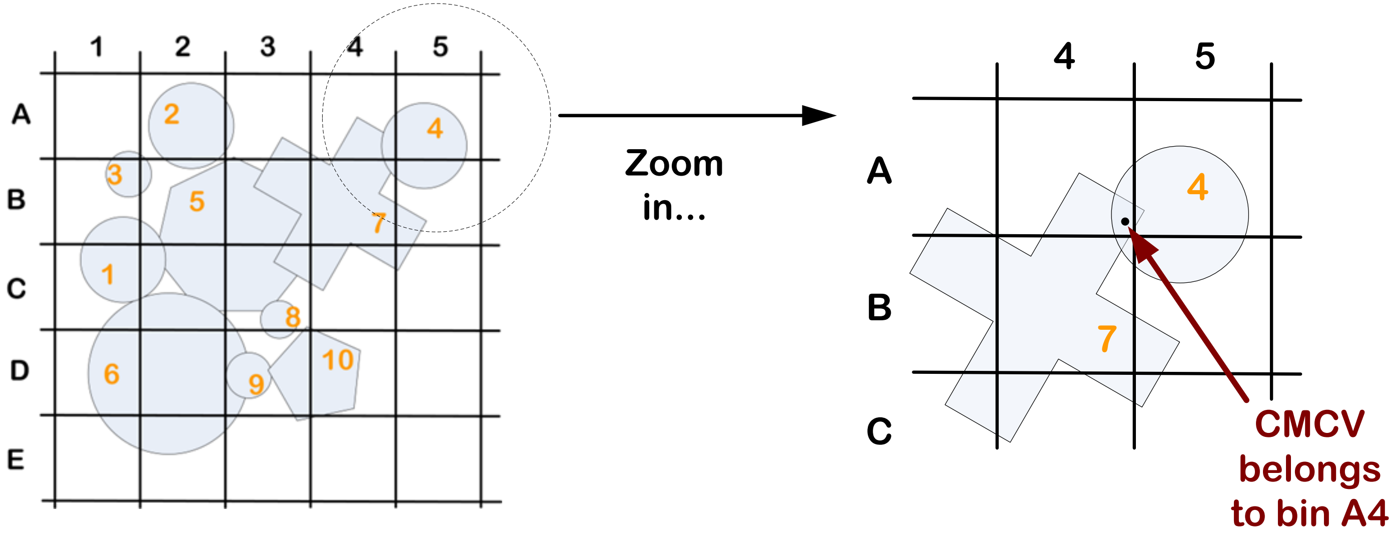
- The drill: thread 0 relies on info at **C**[0], thread 1 relies on info at **C**[1], etc.
- Let's see what thread 2 (goes with **C**[2]) does:
  - Read the first 2 bodies that start at offset 6 in **B**.
    - These bodies are 4 and 7, and as B indicates, they touch bin A4
    - Bodies 4 and 7 turn out to have 1 contact in A4, which means that entry 2 of D needs to reflect this

0	1	2	3	4	…
0	1	0	0	0	…
(A2)	(A4)	(A5)	(B1)	(A3)	…

D-array (Length: N_b)

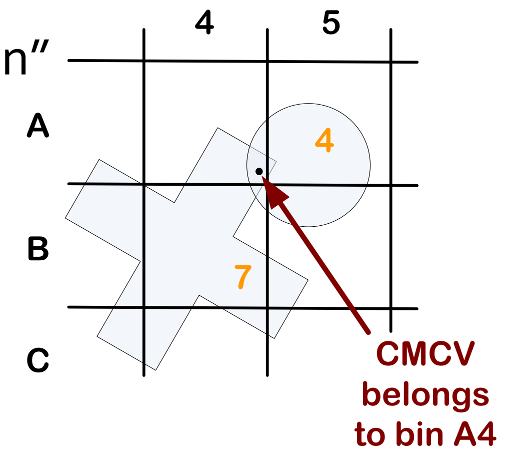
Stage 7: Details…

- Brute Force CD rehearsal
- Carried out to understand the memory requirements associated with collisions in each bin
- Finds out the total number of contacts owned by a bin
- Key question: which bin does a contact belong to?
- Answer: It belongs to bin containing the CM of the Contact Volume (CMCV)



Stage 7, Comments

- Two bodies can have multiple contacts, handled ok by the method
- Easy to define the CMCV for two spheres, two ellipsoids, and a couple of other simple geometries
  - In general finding CMCV might be tricky
    - Notice picture below, CM of 4 is in A5, CM of 7 is in B4 and CMCV is in A4
  - Finding the CMCV is the subject of the so called "narrow phase collision detection"
    - It'll be simple in our case since we are going to work with simple geometry primitives



Stage 8: Inclusive Prefix Scan

- Save to the side the number of contacts in the last bin (last entry of **D**) d_{last}
  - Last entry of **D** will get overwritten

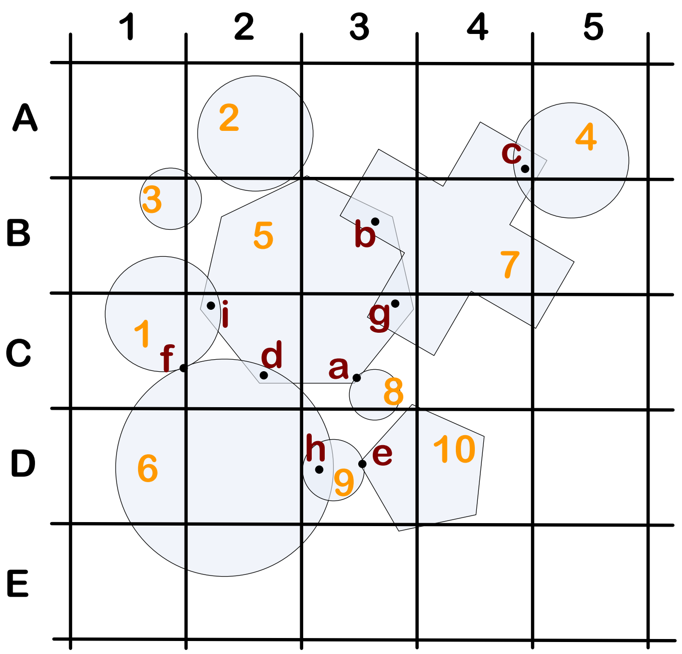
0	1	2	3	4	…
0	1	0	0	0	…
(A2)	(A4)	(A5)	(B1)	(A3)	…

D-array (Length: N_b)

- Run parallel exclusive prefix scan on **D**:

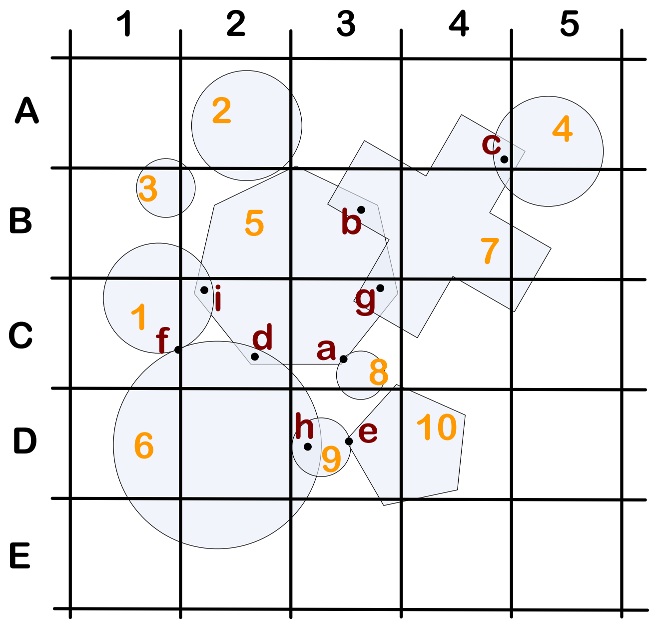
0	1	2	3	4	…
0	0	1	1	1	…
(A2)	(A4)	(A5)	(B1)	(A3)	…

D-array, after exclusive prefix scan



- Total number of actual collisions: $N_c = \mathbf{D}[N_b] + d_{last}$

Stage 9: Populate Array E



- From the host, allocate on the device memory for array **E**
  - Array **E** stores the required collision information: normal, two tangents, etc.
  - Number of entries in the array: N_c (see previous slide)
- In parallel, on a per bin basis (one thread/bin):
  - Populate the **E** array with required info
- Not discussed in greater detail, this is just like Stage 7, but now you have to generate actual collision info (stage 7 was the rehearsal)
- Thread for A4 will generate the info for contact "c"
- Thread for C2 will generate the info for "i" and "d"
- Etc.

Stage 9, details

- **B**, **C**, **D** required to populate array **E** with collision information

0	1	2	3	4	5	6	7	8	9	10	11	
3	2	3	2	5	7	4	7	4	7	1	3	…
A1	A2	A2	A3	A3	A3	A4	A4	A5	A5	B1	B1	…

B-array

Bin vs. Body Touching This Bin

Shows up 2 since there are two bodies (4 & 7) in bin with offset 6 (A4)

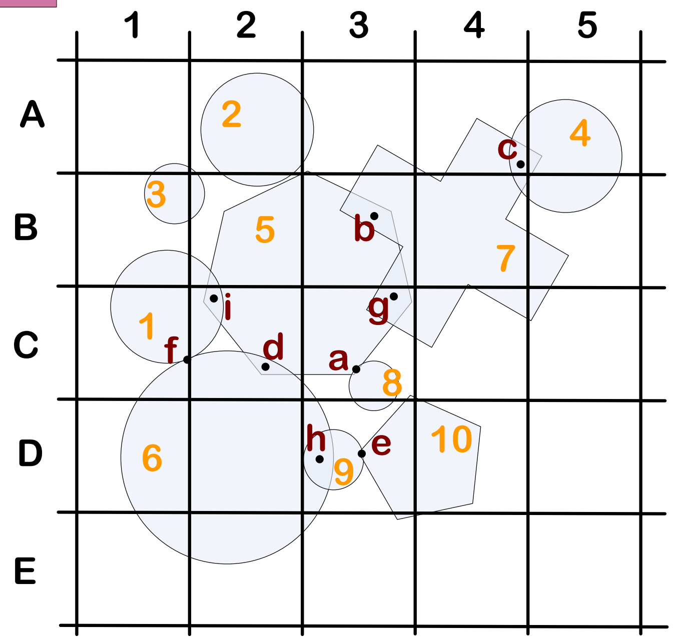
Bin offset in B and number of bodies touching that bin

1	6	8	10	3	…
2	2	2	2	3	…

C-array

0	1	2	3	4	…
0	0	1	1	1	…
(A2)	(A4)	(A5)	(B1)	(A3)	…

D-array, after exclusive prefix scan

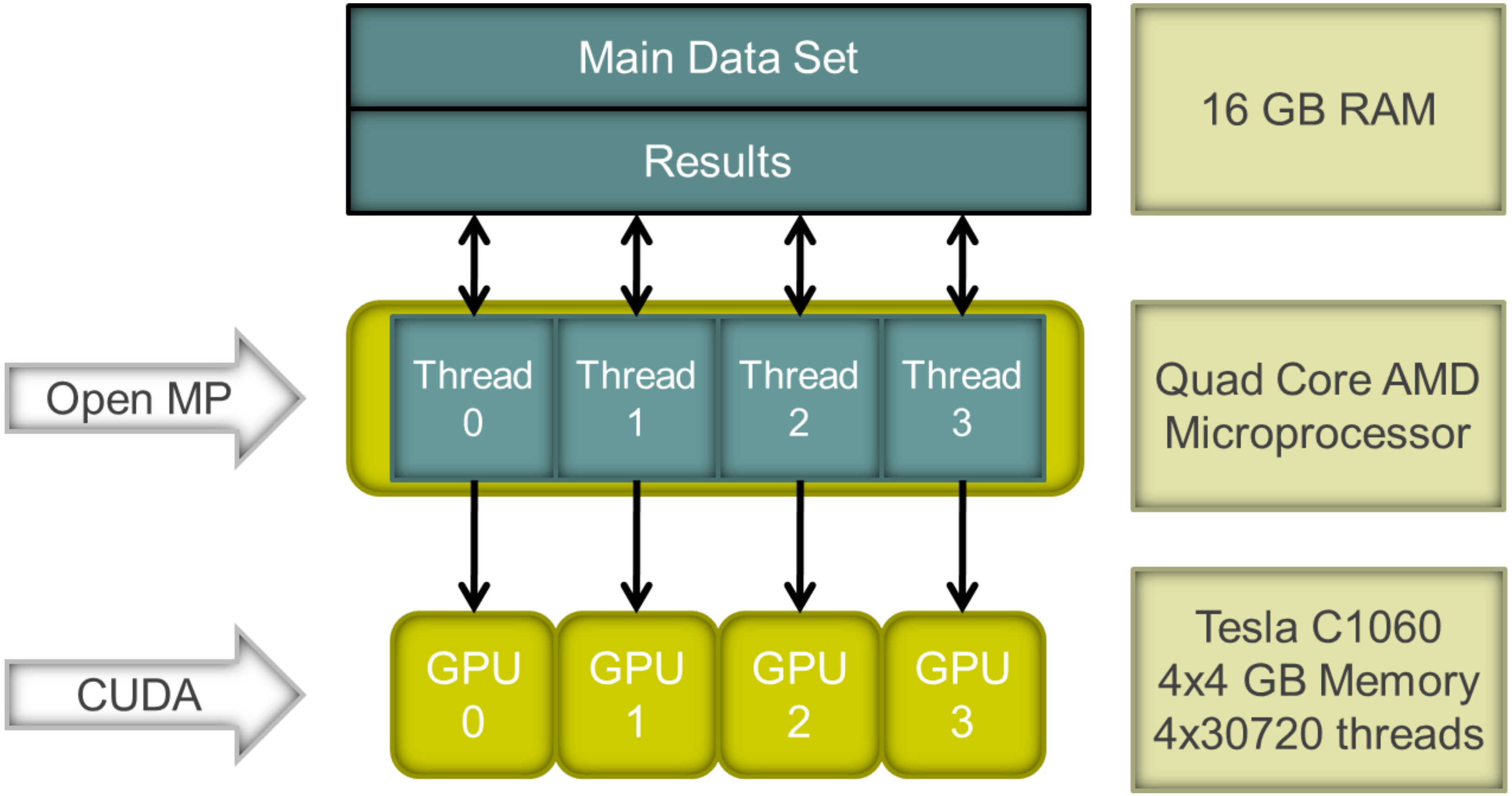


- **C** and **B** are needed to compute the collision information
- **D** needed to understand where collision information is stored in **E**

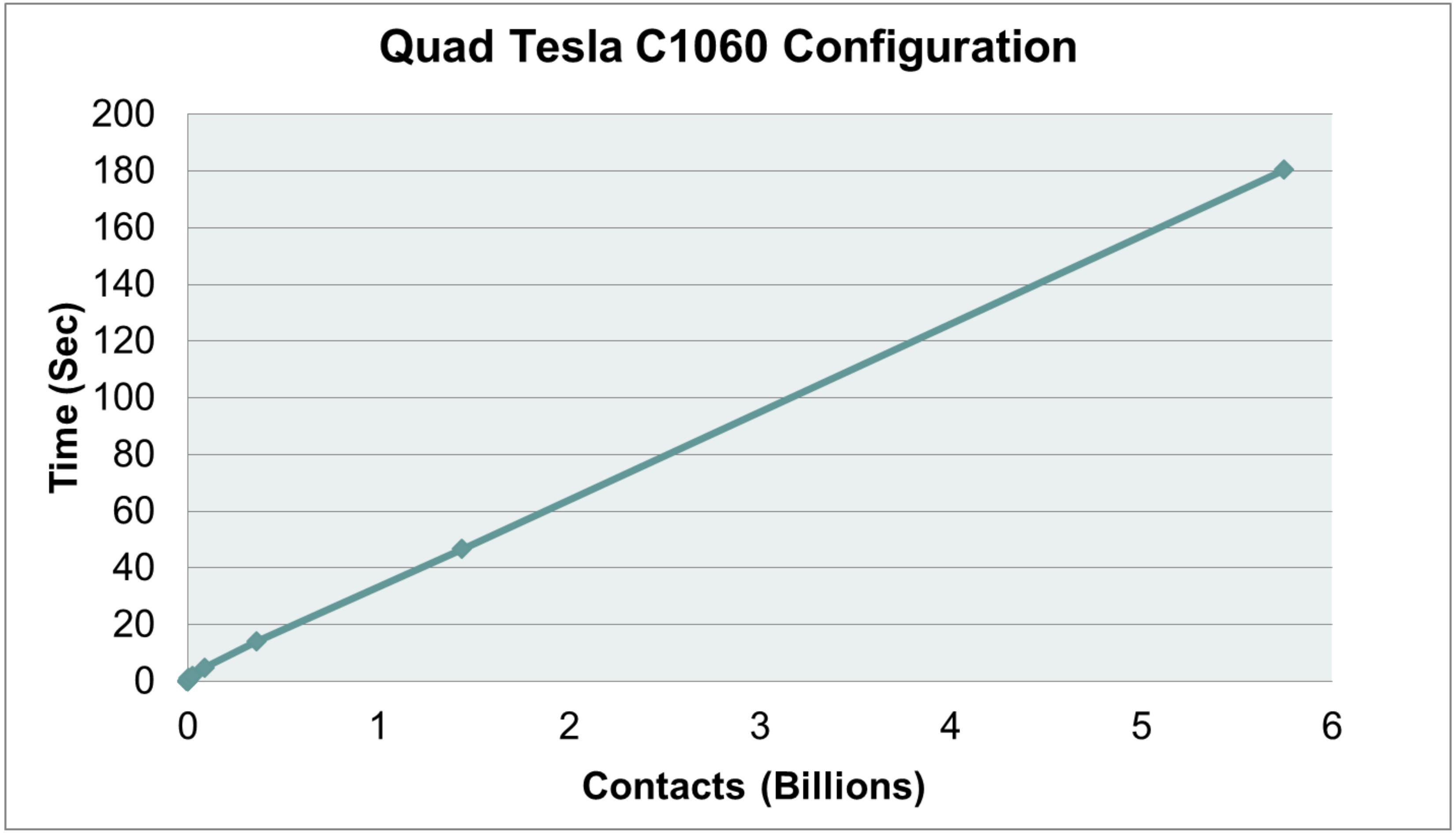
Multiple-GPU Collision Detection

- Processor: AMD Phenom II X4 940 Black
- Memory: 16GB DDR2
- Graphics: 4x NVIDIA Tesla C1060
- Power supply 1: 1000W
- Power supply 2: 750W

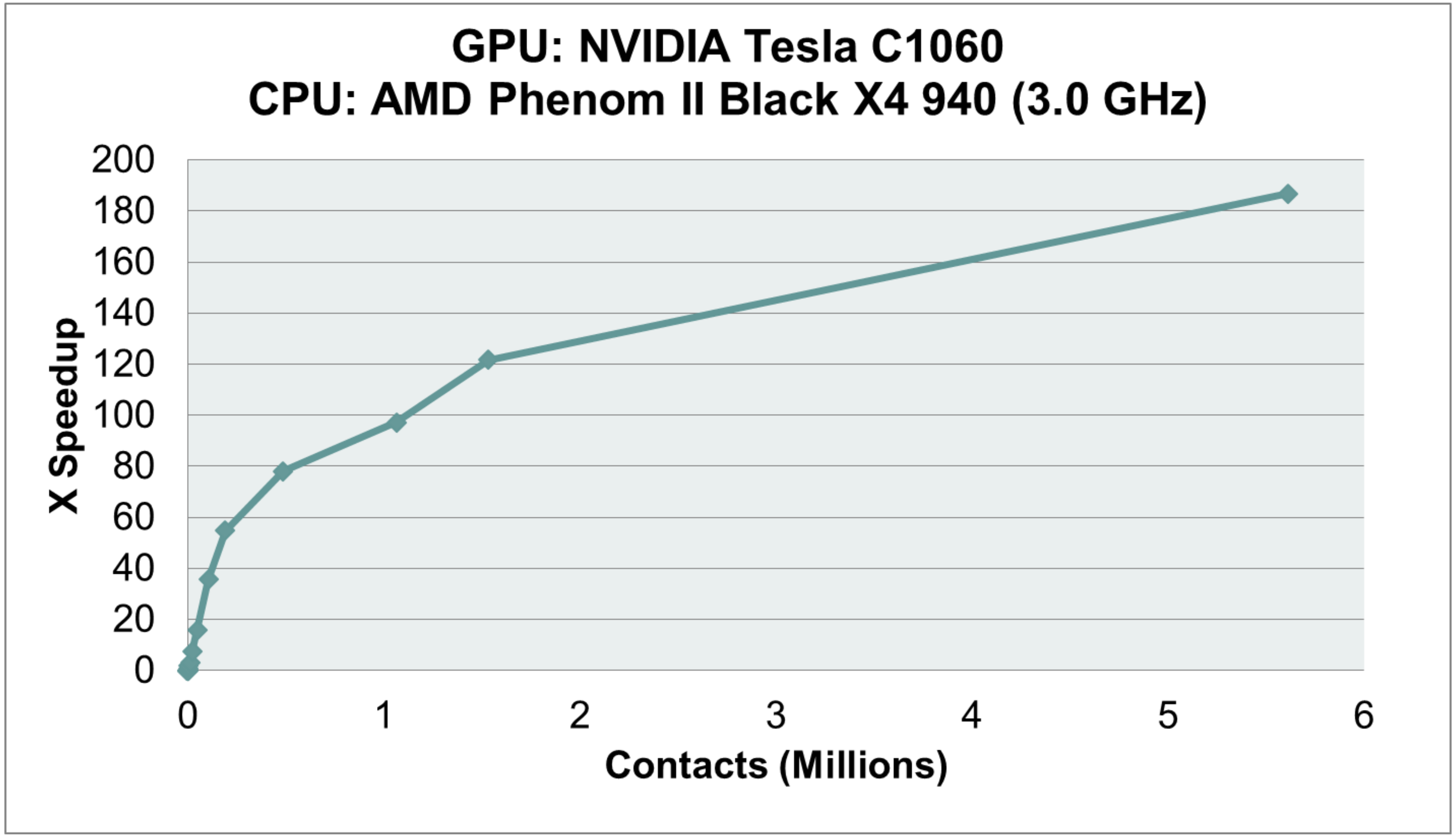
Software/Hardware Setup



Spheres – Contacts vs. Time



Speedup - GPU vs. CPU (Bullet library)
[results reported are for spheres]



Domain decomposition
&
Inter-domain data exchange

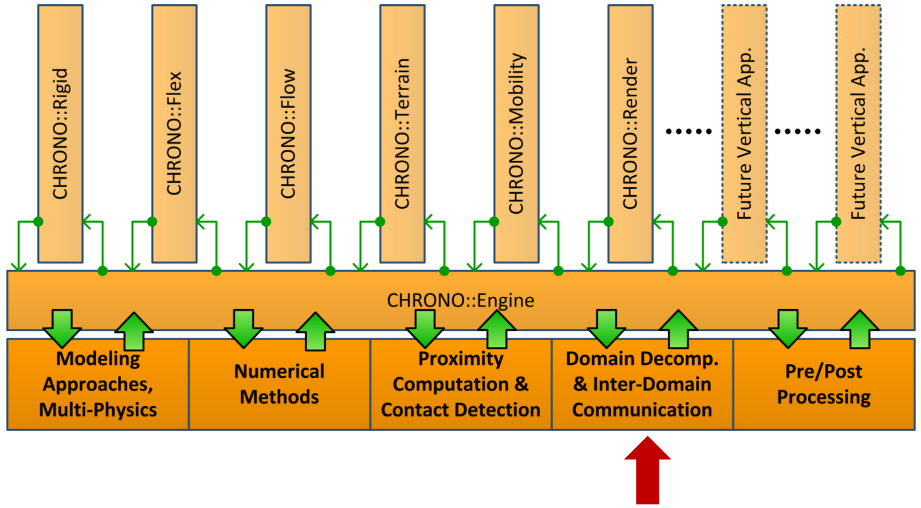
Advanced modeling techniques

Algorithmic (applied math) support

Proximity computation

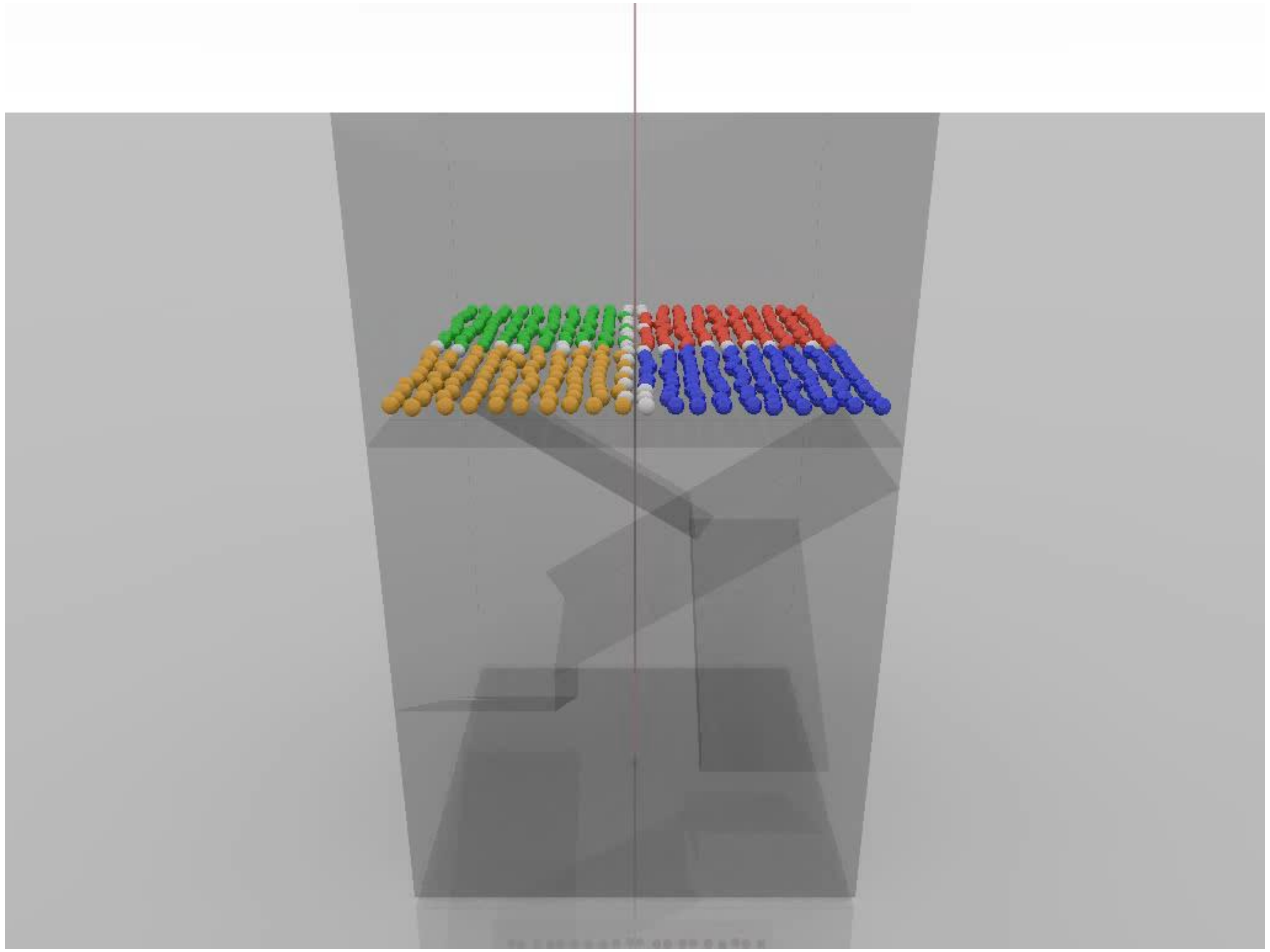
Domain decomposition & Inter-domain data exchange

Post-processing (visualization)



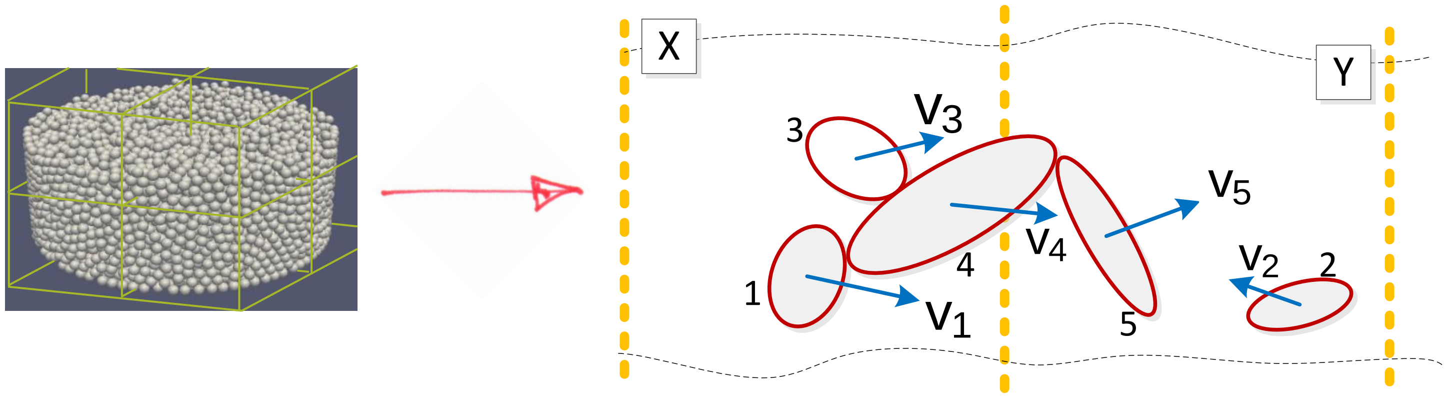
Juggling World Record:
64 People Juggling (of all places) in Madison, Wisconsin

Computation Using Multiple CPUs

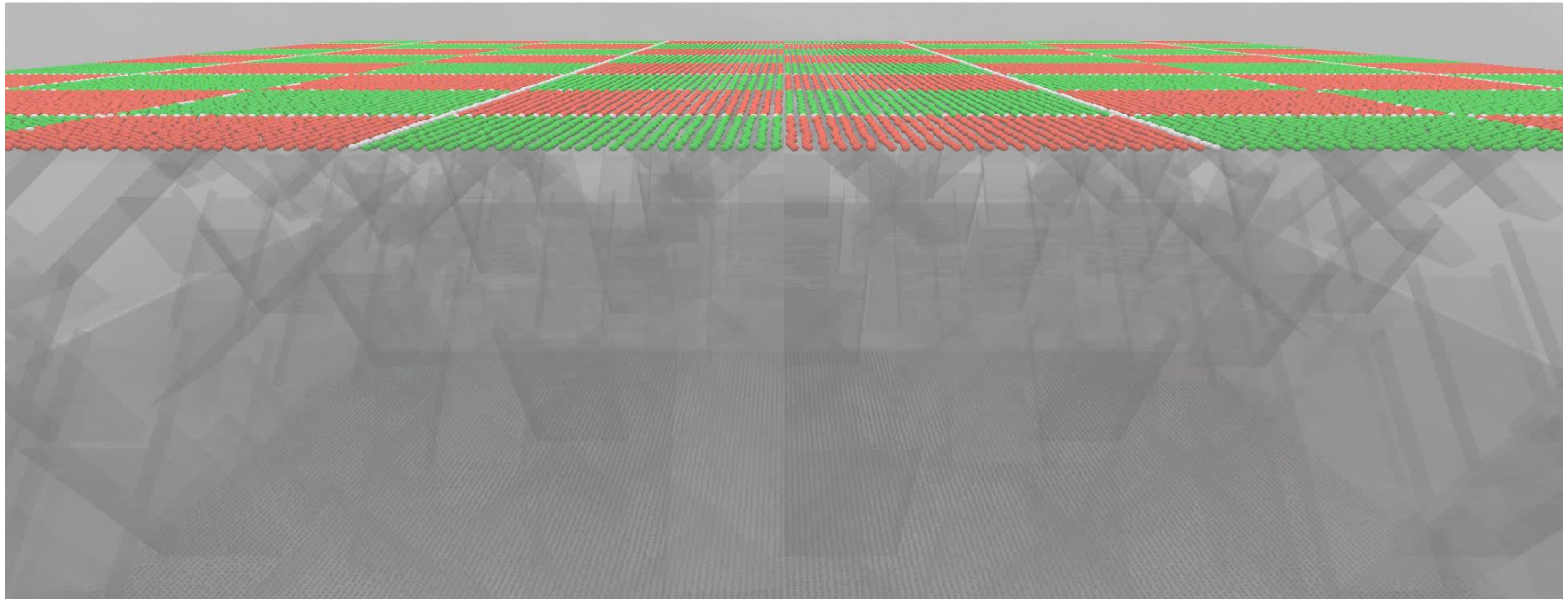


CHRONO:
Domain decomposition & Inter-domain data exchange

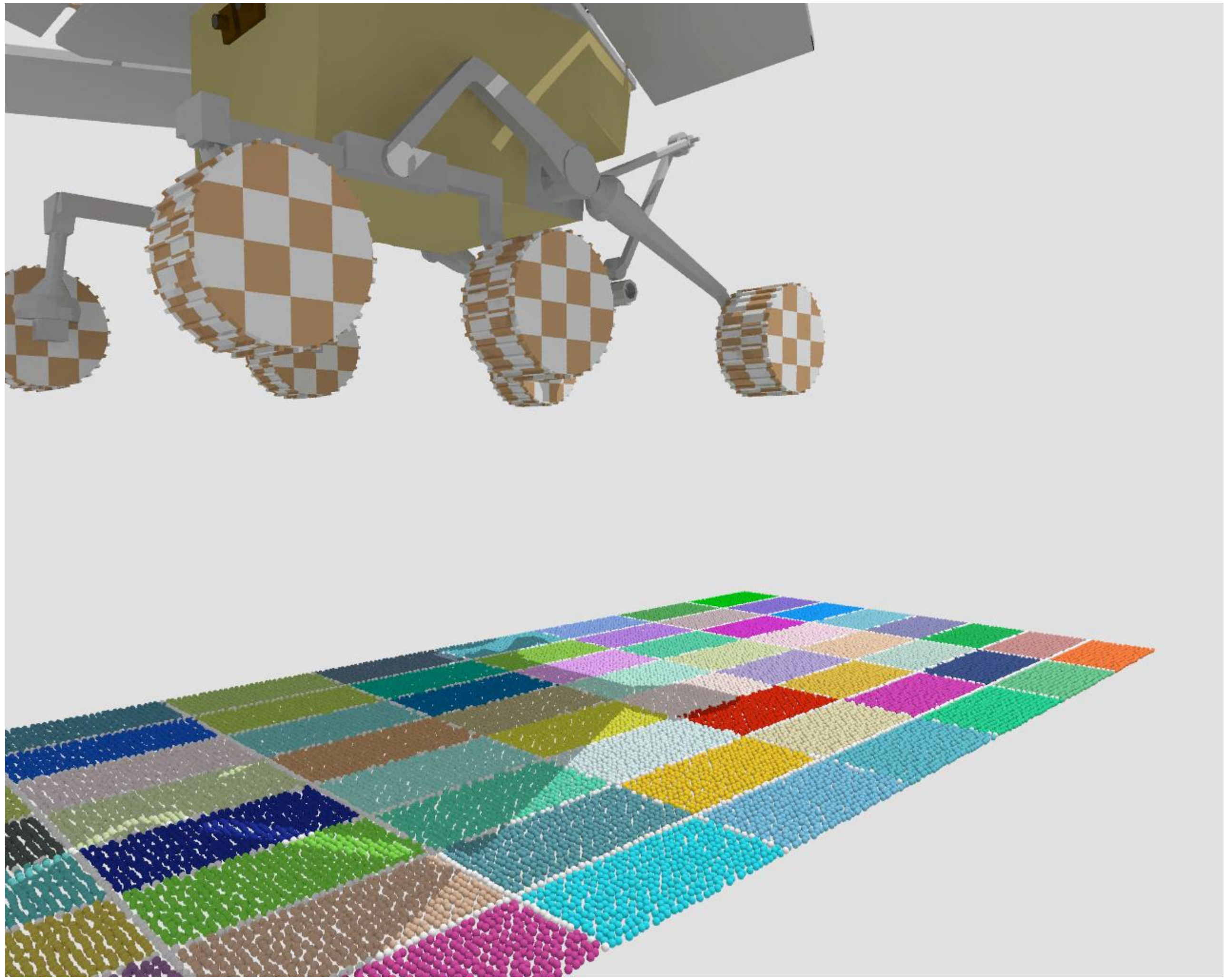
- Divide simulation into chunks and have multiple CPUs/GPUs exchange data during simulation, as needed
- Elements leave one subdomain to move to a different one in transparent fashion
- Key issues:
  - Dynamic load balancing
  - Establish a dynamic data exchange protocol (DDEP) between sub-domains



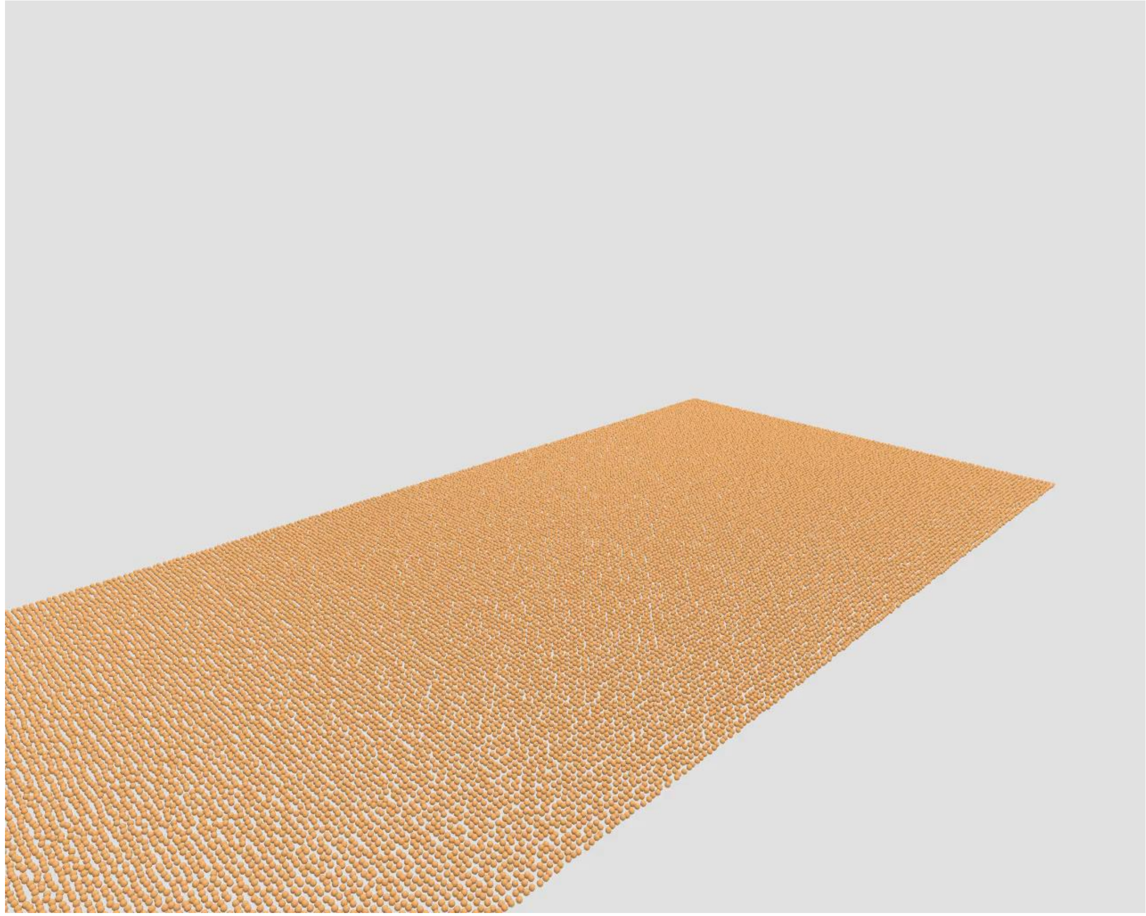
0.5 Million Bodies on 64 Cores
[Penalty Approach, MPI-based]



Computation Using Multiple CPUs



Rover Footprint, Multi-Domain Computation



Pre/Post-processing (visualization)

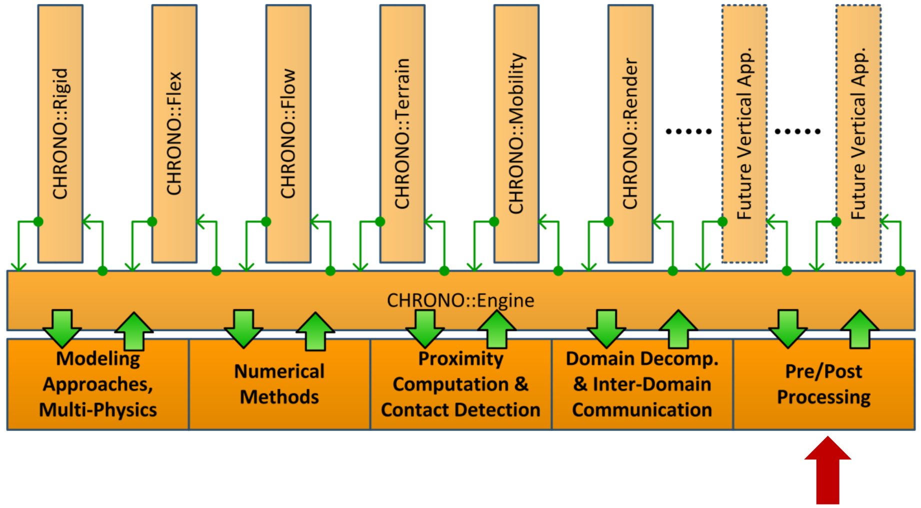
Advanced modeling techniques

Algorithmic (applied math) support

Proximity computation

Domain decomposition & Inter-domain data exchange

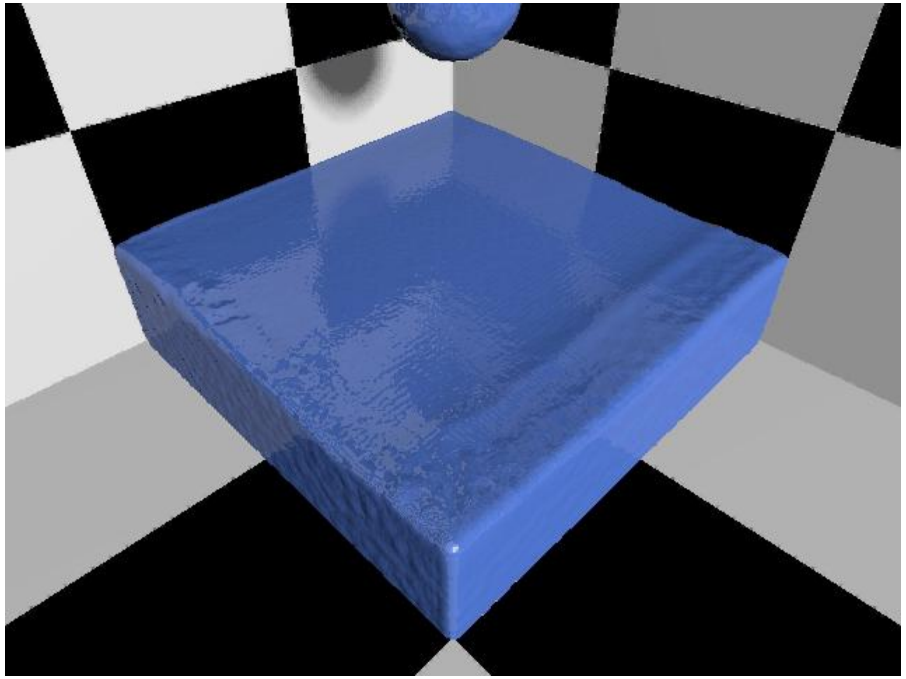
Post-processing (visualization)

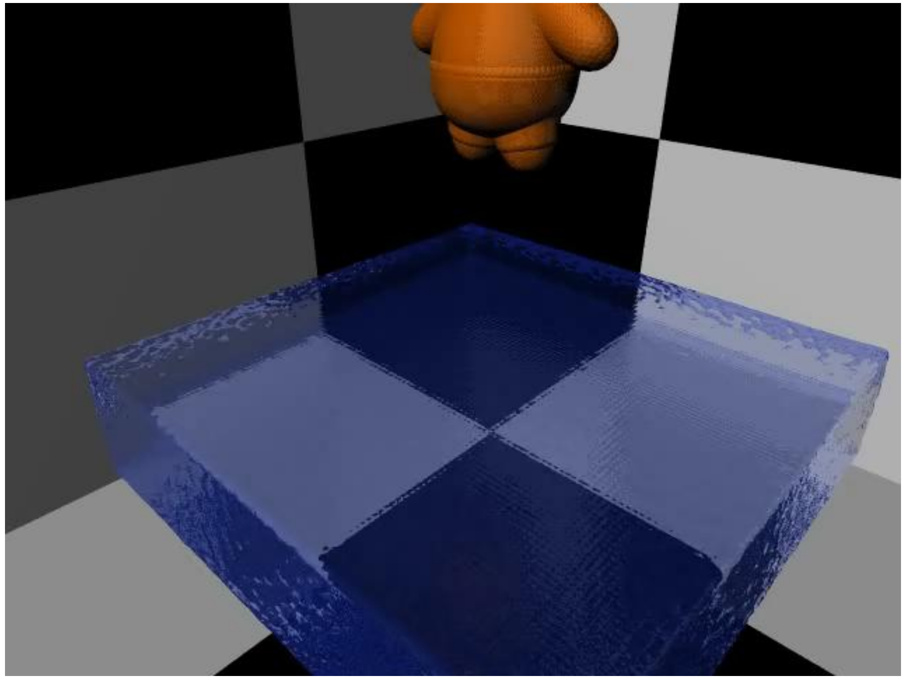


CHRONO: Visualization and Post-Processing

- Rendering very complex scenes with more than one million components
- Rendering takes longer than simulating
- Pursuing a rendering pipeline that leverages parallel computing

Fluid Dynamics and Fluid-Solid Interaction





Rendering Pipeline: Problem Statement

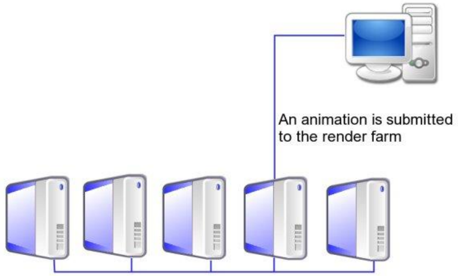
- Render big data: efficiently and beautifully
- Have the flexibility to render anything
- Make the rendering process streamlined and simple
  - Provide rendering as a service

Rendering Pipeline: From Data to Movie

- Example: Tire-Terrain Simulation
  - Data
    - Sequence of Height Maps
    - Render Settings File
  - Cluster
    - Submit data and settings to cluster remotely
    - Schedule jobs and render
  - Movie
    - Returned upon completion

DATA

CLUSTER



MOVIE

UNCLASSIFIED: Distribution Statement A. Approved for public release. #24389

Tire Rolling on Deformable Terrain

Tire Rolling on Deformable Terrain

Rendering Pipeline Uses Pixar's Renderman (PRMan)

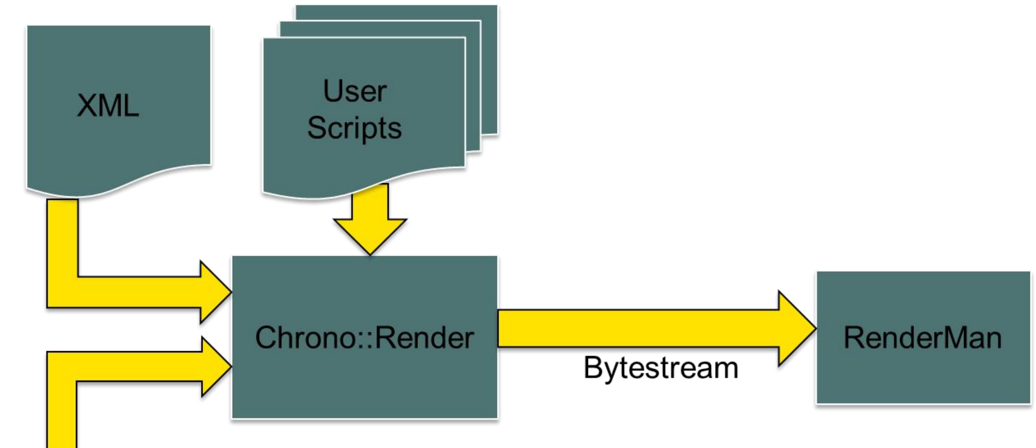
- PRMan: Engineered to be fast, efficient, and configurable for complex scene rendering

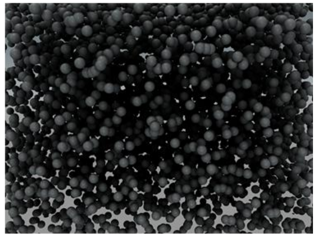
- Pixar's PRMan: industry's rendering standard
  - Lab supercomputer can run up to 320 instances of PRMan

- Open source alternatives:
  - Aqsis
  - JrMan
  - Pixie

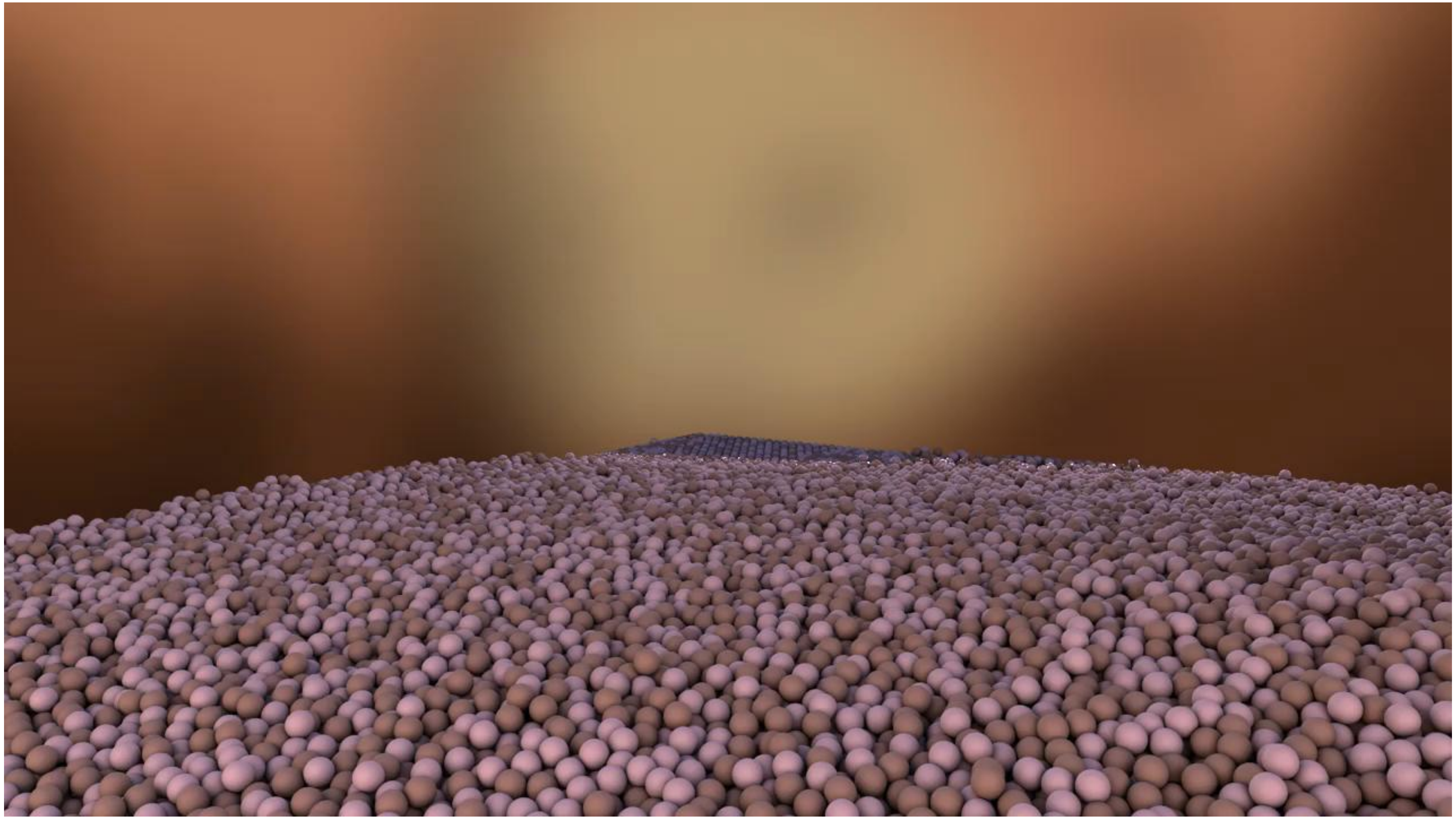
Chrono Rendering Pipeline:

- RenderMan requires a lot of work to configure and optimize correctly
  - Not aimed at science and engineering communities
- Chrono::Render tailored to science and engineering
- Chrono::Render - What is it?
  - C++ binaries, simple Python scripting interface, and succinct XML specification

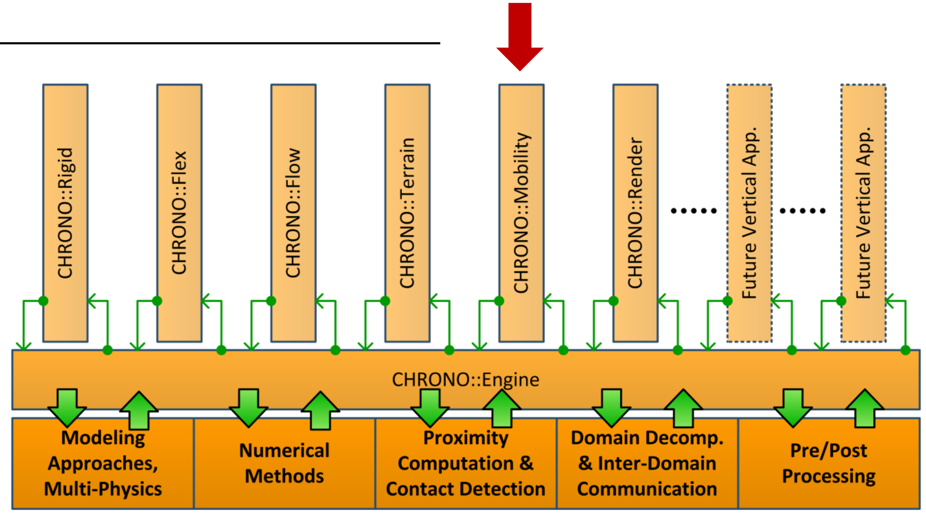




Light Robot Operating on Discrete Terrain



Chrono::Mobility



Terramechanics Modeling Methodologies

1. Empirical methods
   - WES numerics, NATO Reference Mobility Model (NRMM)

2. Semi-analytical
   - Bekker-Reece vertical pressure/sinkage relation
   - Janosi-Hanamoto slip/shear relationship
   - Wong/Reece plastic equilibrium approach

3. Physics-based
   - Finite Element Analysis
   - Particle/Discrete Element methods (DEM, DVI)
   - Meshless/Lagrangian Methods (SPH, MPM, etc.)

Terramechanics for Vehicle Mobility, Remarks

- Empirical methods have limited predictive attributes for general purpose vehicle mobility
- Semi-Analytical methods have been applied to mobility studies with some success
  - See: Trease, Holtz, Azimi, Schmid, Harnisch, Slattengren
  - Limitations due to some (but not necessarily all) of the following assumptions:
    1. Tire geometry is 2-D, circular in shape
    2. Wheel moves forward at a constant velocity and spin rate
    3. Wheel moves parallel to flat ground
    4. Soil is homogenous, perfectly plastic medium
- FEA or DEM are accurate, but computationally expensive
  - Madsen/Heyn/Negrut/Lamb (demonstrated shortly)

Chrono::Mobility – Goals

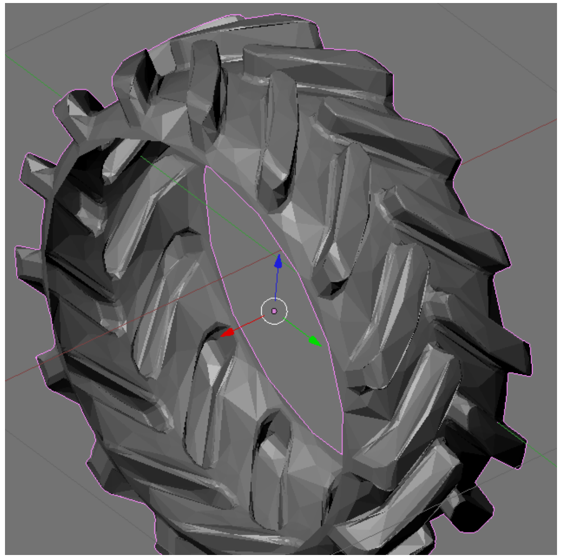
- Develop general purpose simulation capability for analysis of wheeled/tracked vehicle mobility on deformable terrain

a) Handle 3-D tire/track geometry to accurately estimate contact patch size and shape

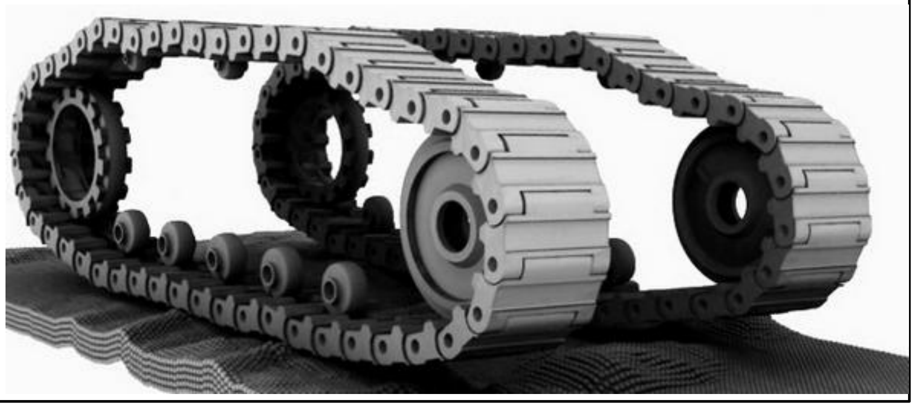
b) Handle general 3-D terrain geometry to allow for realistic mobility scenarios

c) Represent the terrain in a way that considers soil stress state and loading history in a volumetric sense, depending on soil type
   - Cohesive soils – compaction
   - Dry granular soils – shear failure and flow
   - Brittle soils – fracture, shear failure and flow

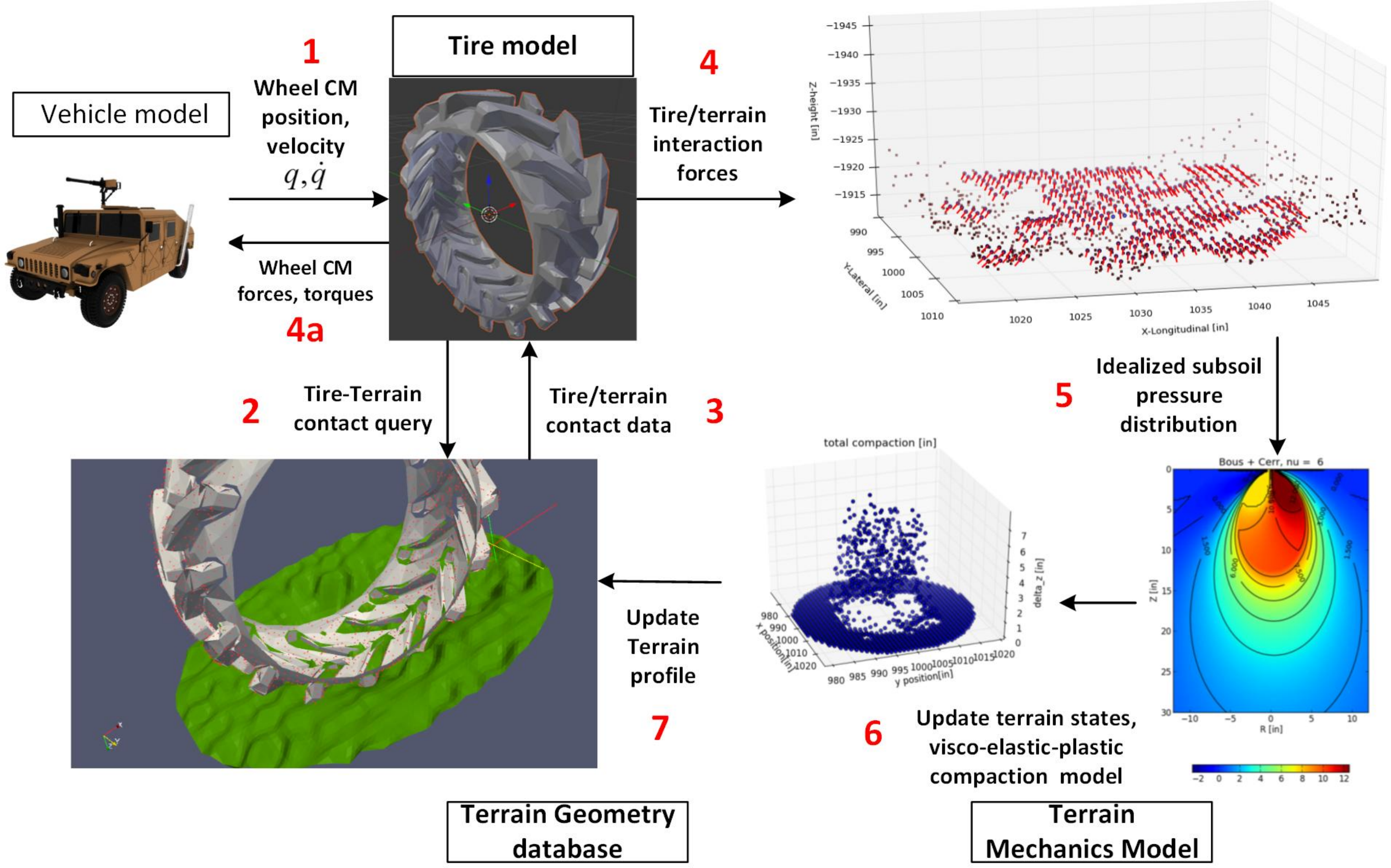
Traction Element Geometry Representation



- 3-D geometry description for tire/terrain collision query
  - Discretized at a resolution to capture tread/lug geometry
- APIs modularized so that terrain database accepts generalized traction geometry when queried
- Directly use Wavefront (.obj) solid models (top)
  - Captures complicated tread/lug geometry
- Can also represent vehicle hull geometry

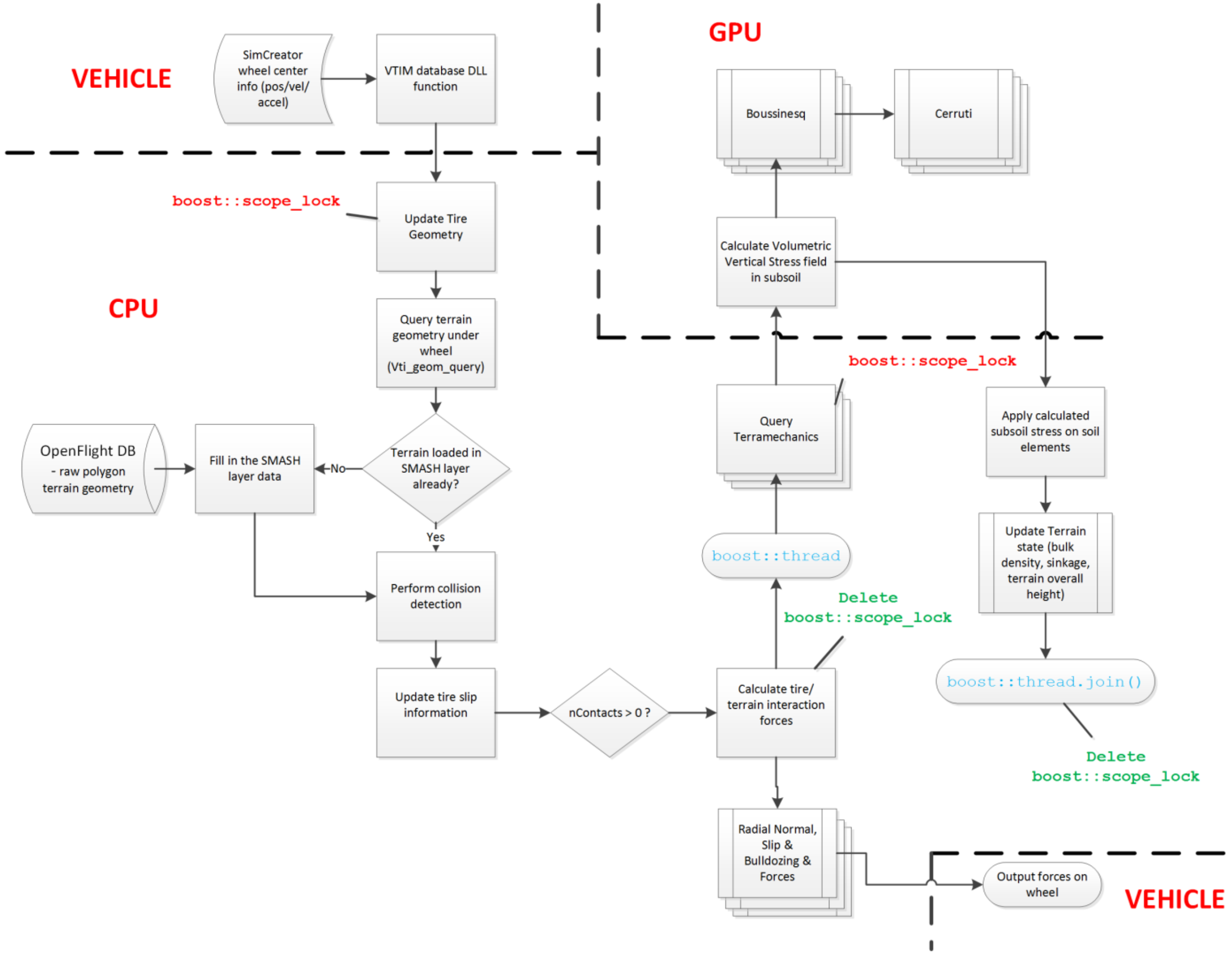


Simulation Framework



Simulation Framework: Implementation Details

- Fast, asynchronous execution
  - Not necessary for vehicle model to wait for query to complete
- Pass STI Force/Moment to vehicle
- Leverages both multi-core and GPU parallelism



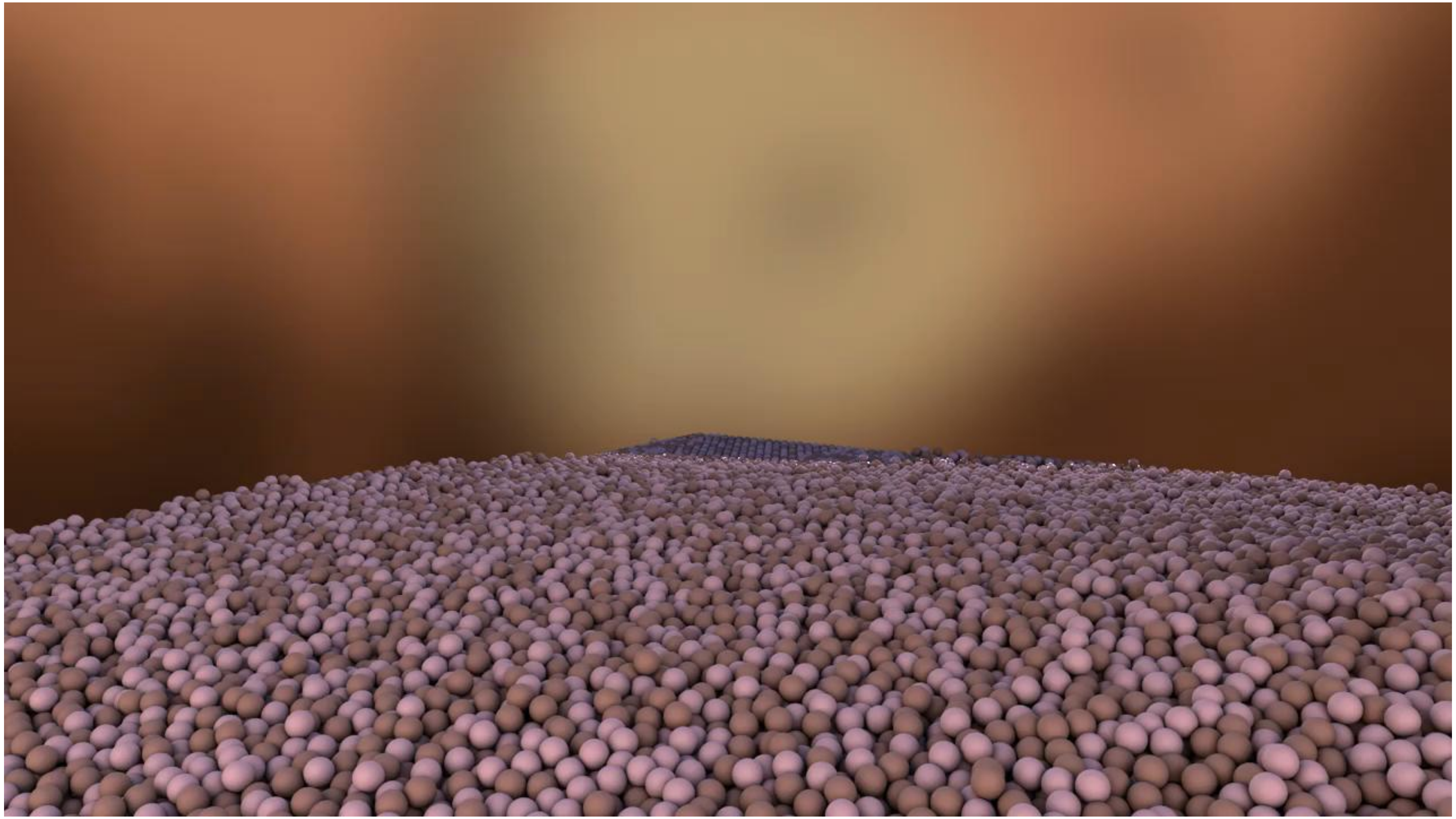
Chrono::Mobility, Today

- Multibody dynamics vehicle model entirely in **Chrono**
  - Generalized 3-D tire/track and terrain geometry representations
  - Visco-elastic-plastic soil model captures soil compaction due to vehicle loads

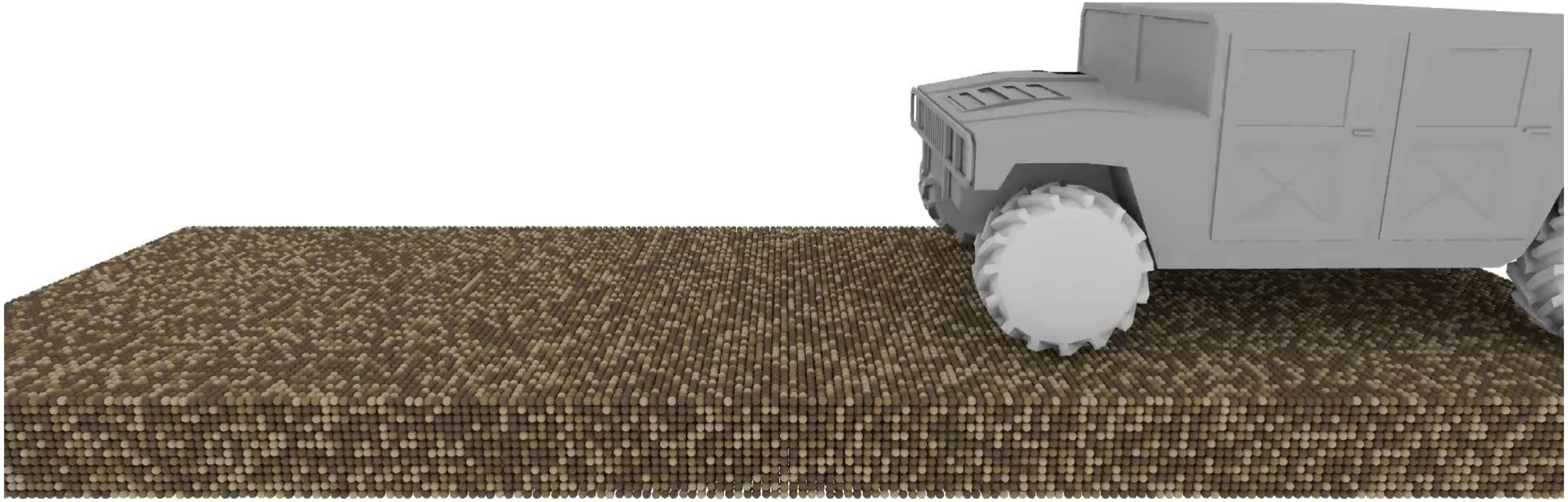
- Leverages other members of the **Chrono** family
  - Based on Standard Tire Interface (STI) & Vehicle Terrain Interface VTI

- Discrete terrain simulation carried out in the same framework

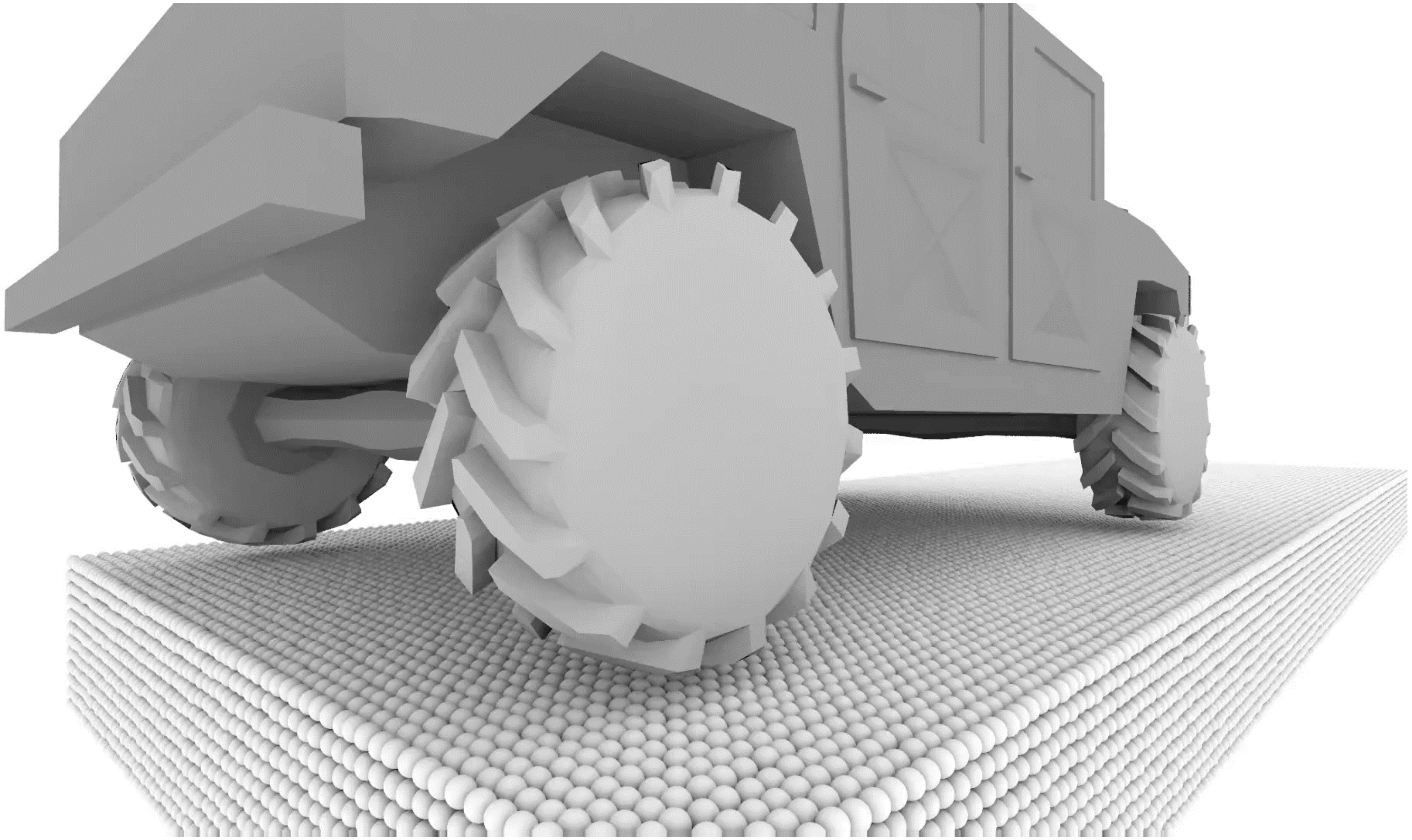
Light Robot Operating on Discrete Terrain



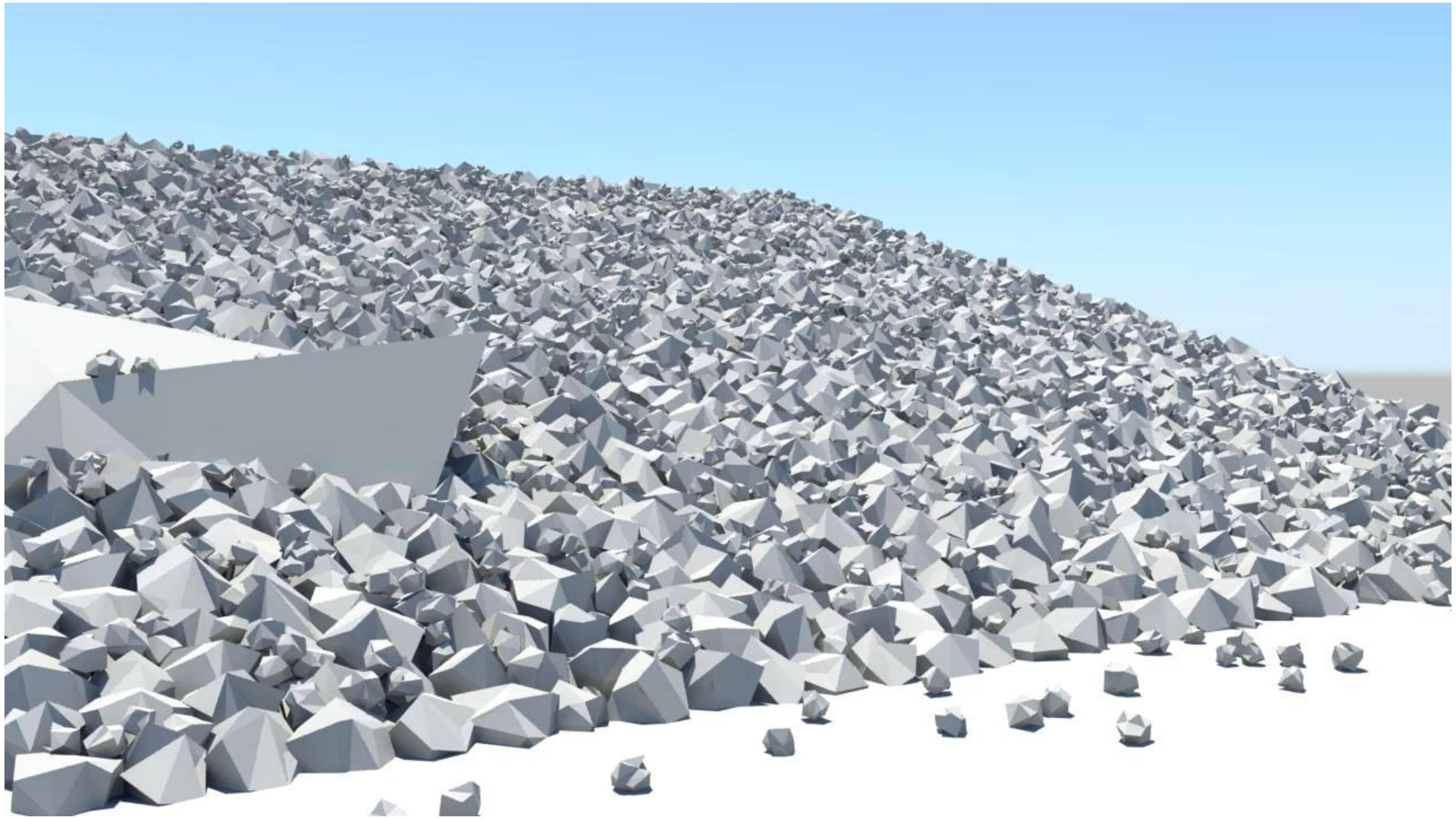
Mobility on Granular Terrain w/ Cohesion



Mobility on Granular Terrain w/ Cohesion: Close-Up

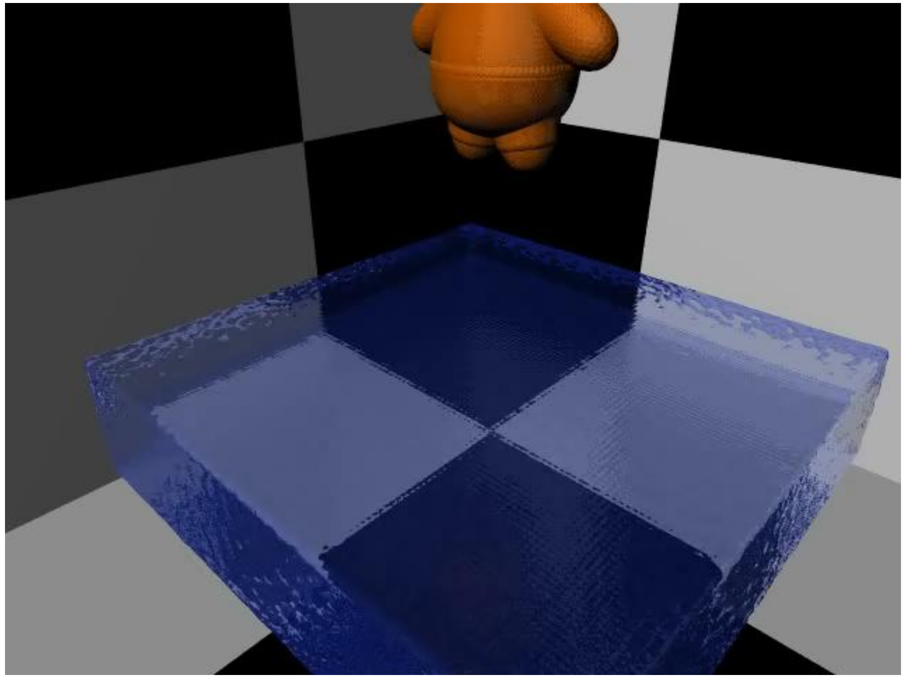


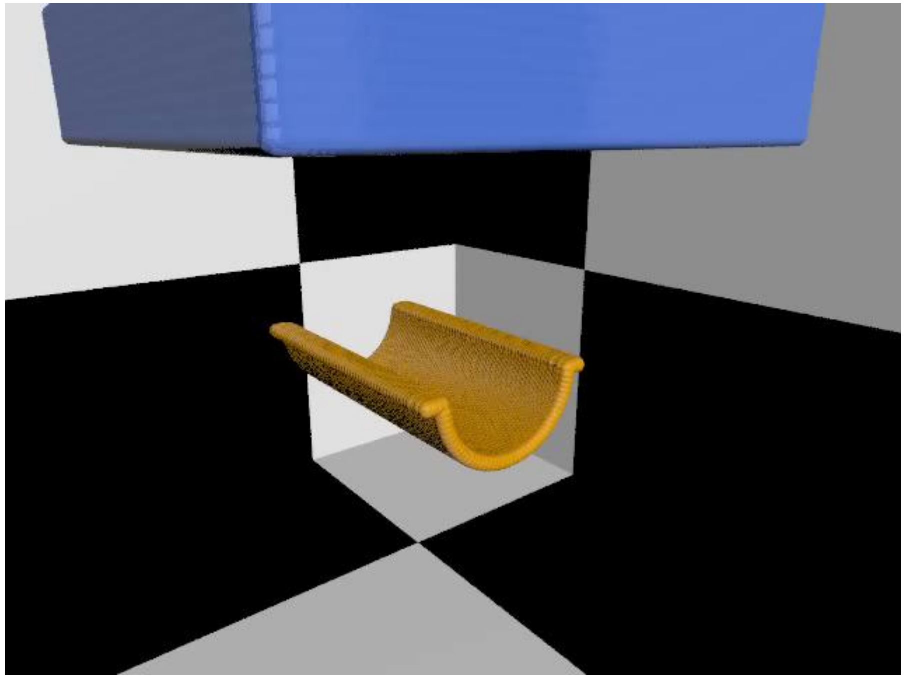
What Can You Do With a Validated Predictive Tool?



Fording, First Cut

Fording, Future Plans: Coupled Fluid-Structure Interaction





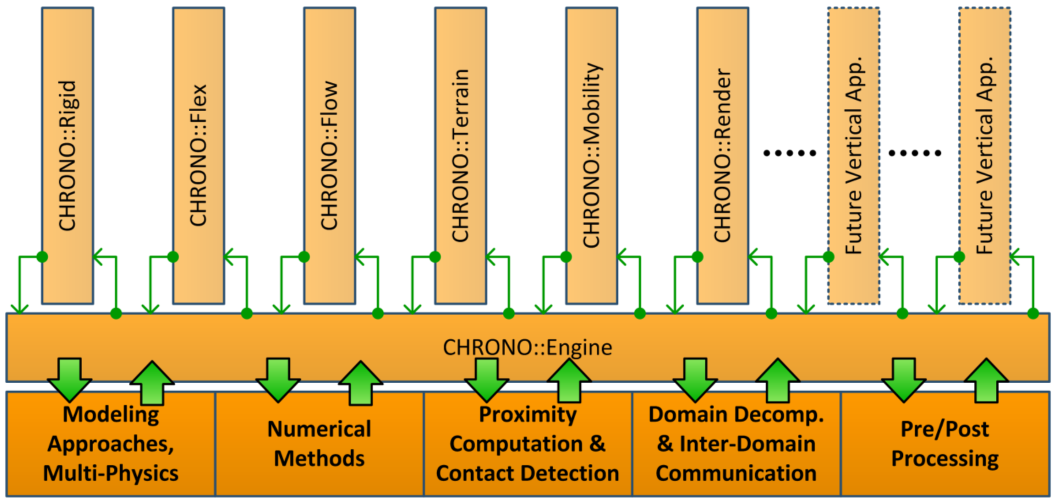
9.3 years of GPU time for simulating Fluid-Solid Interaction problems

Departing Thoughts

Chrono – The Long View

- **Chrono**-effort focused on four thrusts:

  1. Validation – useful
  2. Pre/Post – friendly
  3. New features – versatile
  4. Leverage advanced computing – fast



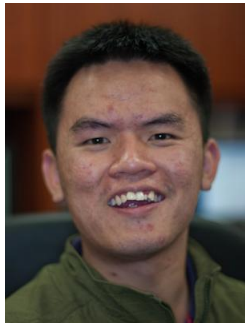
Wisconsin Applied Computing Center

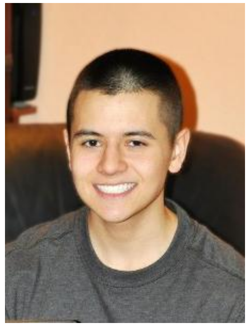
- Our group, Computational Dynamics, has 12 members
  - Three Faculty/Researchers…

- Nine Graduate and Undergraduate Students…








Closing Remarks

- We are focused on physics-based simulation

- Vision:
  - Solve real-world problems (pursue relevant questions)
  - Put computers and good ideas to work
  - Build upon partnerships and collaborations
  - Make outcomes of our work available to users: release early, release often (BSD-3 license)

- Approaching physics-based simulation in a holistic fashion through `Chrono`
  - Modeling + numerical solution + visualization
  - Rely on emerging hardware for fast simulation

Thank You.

negrut@wisc.edu

Simulation Based Engineering Lab

Wisconsin Applied Computing Center

PPT presentation & animations will be available on-line for download.